The Dissertation Committee for Balaji Sampath
certifies that this is the approved version of the following dissertation:

Scheduling and Stability Analysis of Cambridge Ring

Committee:

John Hasenbein, Supervisor

Erhan Kutanoglu

David Morton

Elmira Popova

Sanjay Shakottai

Scheduling and Stability Analysis of Cambridge Ring

by

Balaji Sampath, B. E, M.S.E

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2007

Dedicated to my family and friends.

Acknowledgments

My time at The University of Texas at Austin has been the most memorable period of my life. During the course of my Ph.D I have met people it has been a privilege to know. The foremost among them has been my advisor Dr. John Hasenbein, who fueled my interest in queueing theory with advice, suggestions and encouragement. He gave me a free hand to try many new things and was always there to suggest a new direction to investigate when something did not work out. One of the things I will miss most when I graduate is his sense of humor. I would like to thank him for his confidence in me and his contributions without which this work would not have been possible. The effort he put into correcting my dissertation and helping to make it rigorous will never be forgotten.

I would like to thank my parents, my sister and my fiancee whose confidence in me has never wavered as I have progressed towards my degree. They were there to encourage me and keep me focussed whenever I needed them.

I have also been blessed with great friends throughout my life. Some of my best friends are from Austin and I would like to thank every one of them for helping me negotiate my Ph.D. In particular I would like to thank Amit Partani for his help. He was always there when I needed to test out my

work before I put it to Dr. Hasenbein. He has been a great asset offering me advice and I will miss his earthy sense of humor and caustic comments when he felt that I was not being productive when we were working together. Navin Varadarajan and Krishna Lakshminarasimhan are two people from whom I learnt the meaning of the word work ethic. They were the people to whom I poured out my woes most of the time. I would also like to thank my high school friends Venkatesh Tanuku and Naresh Rao Bhagavatha who currently live in the Greater Austin area. They were there for me whenever I needed a break, checked on my progress whenever I had a deadline to meet and were there to keep me focussed. Sharadha and Hariharan Kalyanaraman also deserve my thanks in no small measure for all their attention and kindness to me over the many years I have known them.

Also I owe my gratitude to John Hall for his help in "digitizing" all my class notes and research notes. John and Suzanne are also personal friends and I have enjoyed spending many an evening with them. Jagan Rajan, Valli Shanmugham, Ramakrishnan Viswanathan, Siva Srinivsan, Vivek Balaji Padmanabhan, Karthik Ramachandran, Karthik Kalyanaraman, Titash Sridharan: You guys have been great friends and my dissertation would be incomplete without gratefully acknowledging your help and advice.

I would like to thank Ruth Schwab, the graduate coordinator of the Mechanical Engineering Department for her help through my years at UT Austin. She made the process of working at the university so much smoother. Rarely have I met a person who enjoyed their work more.

Scheduling and Stability Analysis of Cambridge Ring

Publication No. _____

Balaji Sampath, Ph.D.
The University of Texas at Austin, 2007

Supervisor: John Hasenbein

Multiclass queueing networks are widely used to model complex manufacturing systems and communication networks. In this dissertation we describe and analyze a multiclass queueing network model known as the Cambridge Ring. As the name suggests this network has a circular topology with unidirectional routing. In many cases the analysis of a stochastic model is a difficult task. For a few special cases of this network we show that all non-idling policies are throughput optimal for this system. One of the major differences between this work and previous literature is that we prove throughput optimality of all non-idling policies, whereas most of the previous work has been on establishing throughput optimality for a specific policy (usually First-In-First-Out).

We use a macroscopic technique known as the fluid model to identify optimal policies with respect to work in process. In one case we consider, the discrete scheduling policy motivated by the optimal fluid policy is indeed optimal in the discrete network. For the other special case we show by means of a

deterministic counterexample that the discrete policy most naturally suggested by the fluid optimal policy may not be optimal for the queueing network. We also formulate the fluid holding cost optimization problem and present its solution for a simple version of the Cambridge Ring. Further we establish that the optimal policy under a class of policies known as "non-ejective" policies may be an idling policy. We use an example of the Cambridge Ring with a single vehicle to show that the optimal policy for this example has to be an idling policy.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Large manufacturing systems are in general hard to model accurately due to complex interactions between arrival and service activities and difficulties arising from various sources such as randomness and complex routing schemes. Some of the techniques available to study the performance of these systems are:

- Deterministic planning techniques such as job shop scheduling
- Simulation
- Stochastic models such as multiclass queueing networks.

The disadvantage of deterministic planning techniques is that they do not always adequately represent varying conditions such as arrival rates or machine breakdowns. In addition, even when there is perfect information on future arrivals and other nominally stochastic events in the system, deterministic optimization in most large systems leads to a combinatorial problem which is NP-Hard.

Simulations provide an inexpensive way of testing the performance of a system. But again, for most large systems, simulation is computationally intensive. In

addition simulation may be used to test the performance of the system given certain parameters, but it cannot for example, provide us with exact conditions under which a system is stable. The number of options that can be exercised in the control of such systems is limited.

Multiclass queueing networks (MQNs) have been used to model manufacturing systems and telecommunication networks. Due to their flexibility, they can be used to represent a wide variety of systems such as job shop scheduling systems or local area networks. One of the advantages of modeling a system as a MQN is that it provides us with a method of analyzing the stability of the system using fluid models. We can also use the fluid models of these systems to develop heuristic policies that try to optimize various objectives such as draining time, holding cost or work in process (WIP). These fluid models treat job flows as a continuous deterministic process and permit us to optimize the system with respect to the performance measures mentioned above.

The main drawback of this method is that the optimization problem is still extremely hard to solve except for a few simple cases. It is to be noted that the analysis of a MQN suffers from other drawbacks such as the fact that the transient behavior of these models is often intractable. Furthermore, even steady state distributions are tractable only under restrictive conditions on the primitive model distributions. Hence these models have limited use in optimization of objectives such as holding cost. In this dissertation, we model a few special cases of a material handling system using multiclass queueing networks. We then proceed to use fluid models of this system to analyze its

stability and to develop policies which optimize work in process.

The queueing network model which we have chosen to analyze is known as the Cambridge Ring (CR) model. The model derives its name from a circular local area network (LAN), originally designed at Cambridge University in the late 1970s. This LAN was designed to link computers to enable data transfer at rates of up to 10 Mbps.

The CR model can be used to represent a number of systems with reasonable accuracy. For example, the Automated Material Handling System (AMHS) in many 300mm semiconductor wafer fabs can be represented using this model. Vehicular traffic in circular loops can be represented using the CR model. A third motivation is to analyze the performance of the local area networks which provided one motivation for the model. A brief description of these applications and potential benefits to be realized are described in section 1.2.

1.1 Problem Description

In this section we provide a high level description of the CR. In section 2.1, we give a detailed description of the mathematical model. The network consists of N stations. The stations are arranged in a ring and numbered $1, \ldots, N$ clockwise as shown in figure 1.1. In addition to processing stations, the CR also has N or fewer vehicles which move clockwise around the ring. A part of our work has been devoted to systems with less than N vehicles. At each station jobs wait in buffers to be loaded onto an empty vehicle when it arrives at the station. We assume that the travel time between stations for all

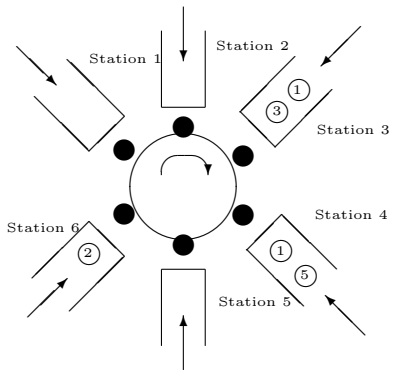

Figure 1.1: The Cambridge Ring

vehicles is constant. We normalize this travel time to one. Hence the whole ring of vehicles moves one unit clockwise every unit of time. At each station there are $N-1$ potential job classes arriving. Each class of jobs requests a particular destination along the ring to which it must be transported. We assume that jobs request travel times less than one full rotation around the ring. Hence there are a total of $N \cdot (N-1)$ potential job classes. It is assumed that the external arrival processes satisfy a strong law of large numbers type assumption.

Each vehicle can carry one job at any time. A vehicle arrives in front of a station at each integer point of time. When a vehicle discharges a job at a particular station, it may pick up a job waiting for service at that station or travel to the next station empty depending on the operating policy. An example of a Cambridge Ring with six stations is shown in Figure 1.1.

1.2 Applications

1.2.1 AMHS Applications

With the demand for digital devices ever on the increase, worldwide semiconductor revenues have been climbing since the late 90s. The number of fabs in operation has also gone up correspondingly. The initial investment and subsequent operating costs of these wafer fabs are very high. If the cycle time of a fab can be decreased by a small percentage it could mean savings on the order of millions of dollars per year. An AMHS is almost indispensable in the newer 300mm fabs for several reasons. The production process could include hundreds of steps with jobs going through complex routes. Human error in transporting the wafers could lead to considerable losses in damages. Another reason is that the lots (usually cassettes of 25 wafers) are too heavy to be routinely transported by human operators. The use of an AMHS provides the opportunity for optimizing costs by using improved scheduling rules. It will be seen that sometimes simple policies, such as FIFO, can be far from optimal for a system.

Further an important component of costs to be considered in any business is the inventory cost associated with the WIP. Average WIP is related to the average cycle time by Little's Law. Almost all businesses try to minimize the average level of the work in process. WIP is however necessary because it provides a buffer against variability and also helps reduce setup costs. Hence some of the interesting questions that arise when operating the AMHS system are:

- Which is the optimal policy with respect to holding costs?
- Which is the optimal policy with respect to makespan?
- Could an idling policy perform better than a non-idling policy with respect to a performance measure such as work in process?

Simulation studies of a CR system with 6 stations performed by Bauer [4] show a wide variation in cycle times among different policies. The average cycle times found are listed in Table 1.1. The arrival processes to each of the stations were independent Poisson processes and the job destination requests were from a discrete uniform distribution. The arrival rate was the same across all stations and the utilization of each station was set at 95%. The results are averaged over twenty simulation runs of 110,000 units with a warm-up period of 10,000 units. The scheduling policies tested [15] were:

- First In First Out (FIFO): Jobs are served in the order of arrival to the station.
- Shortest Requested Travel Time (SRTT): The job with the shortest travel request receives highest priority.
- Longest Requested Travel Time (LRTT): The job with the longest travel request receives highest priority.
- MOST-CHOICES: The job whose destination station has the most number of job types currently waiting is served first. If there are multiple jobs

Scheduling Policy	Mean Cycle Time
Shortest Requested Travel Time (SRTT)	43.37
Longest Requested Travel Time (LRTT)	26.90
FIFO	24.18
Most Choices	20.27
MaxWIP-SRTT	19.77
MaxWIP-FIFO	19.08
MaxWIP- LRTT	18.91

Table 1.1: Cycle Time Simulation Results

whose destinations have the same number of choices, ties are resolved using LRTT.

- MAXWIP-*: The job whose destination queue has the largest total number of jobs currently waiting is served first. Again a tie-breaking rule such as FIFO or SRTT is required. * represents this rule.

These simulation results indicate that even in a small system, a good choice of scheduling policy can result in substantial savings in terms of mean cycle time, and hence by Little's Law, the average WIP.

Analytical models of actual AMHS systems tend to be fairly complex and hard to use. The most prevalent technique for analyzing these complex systems is simulation. As is evident from the problem description provided in section 1.1, a CR model can be viewed as a simple AMHS and hence the analysis of CR model could yield valuable insight into optimal scheduling of the AMHS with respect to various performance measures.

1.2.2 Network Applications

As mentioned earlier the CR derives its name from a circular LAN. Analyzing the stability and scheduling aspect of CR type LANs using the newer developments in queueing systems such as fluid models or diffusion approximations could lead to substantial improvement in terms of performance measures such as delay.

1.3 Literature Review

Some of the earliest analysis on this problem dates back to Avi-Itzhak [2]. Under the assumption that no job travels more than one full rotation around the ring, he analyzes the traffic behavior for a CR under heavy loading. Heavy loading is assumed to occur when the traffic intensity at each station is equal to one. This leads to a deterministic mathematical programming problem for describing the traffic flows. He also describes an algorithm to solve this problem and hence determine the traffic flows. The system is assumed to be operating under the FIFO policy at each station.

A substantial amount of research has been done on the LAN applications of the CR. The properties of CR type LANs and their effect on its uses were described by Needham [20]. This work focuses on some of the physical implementation aspects of the LAN such as the probability of damage during transmission and the timing requirements which affect the transmission protocols. In a subsequent paper, Hopper and Needham [16] discuss the architecture, implementation details and transmission protocols for the Cambridge Fast Ring

Networking System. King and Mitrani [17] modeled the CR type LAN, studied the performance of different protocols, and compared its performance to an alternate ring configuration known as the token ring network.

Dantzer and Dumas [11] model the CR as a discrete time Markov chain and derive exact stability conditions. They also develop a fluid model of the system and, using a Lyapunov function approach, prove that the system is stable under the usual traffic conditions (UTCs, see section 2.1.2) when the system operates under the FIFO discipline. They also demonstrate that even in a simple system the fluid limits exhibit unusual behavior.

Coffman et al. [13] also investigate the stability of the CR under the FIFO policy at each station. They also assume independent and stochastically identical arrivals to all the stations. Under these conditions they demonstrate that, as long as no customer requests a travel length more than a single rotation around the ring, a sufficient condition for stability is that the total arrival rate to the system is less than one. They also present simulation studies that indicate that the system is stable as long as total arrival rate is less than two if the requested travel distance of jobs is uniform on $\{1, \ldots, N-1\}$. They also study the asymptotics of the system when the number of vehicles tends to infinity.

Coffman at al. [12] in a subsequent paper develop an approximation to this system when the CR is stable but very long queues of jobs form at the stations. They accomplish this by assuming that every time a job is discharged at its destination station, there is a job waiting to be picked up at that station. In

that paper the distribution of times between successive deliveries is derived for a general distribution of job transit times. They also show that for a large number of vehicles this distribution is approximately exponential when job destinations are chosen uniformly.

In most of the work discussed above, the fact that system managers might know the destinations of the jobs waiting in the queue has not been used. In other words the general approach has been to analyze the problem as if the vehicle were a taxicab and the cab driver does not know the destination of a customer until he or she boards the cab. We would like to take a more nuanced approach in that we know the destinations of jobs waiting in the queues and use this information in scheduling.

Bauer [4] develops efficient scheduling algorithms for the CR when job destinations are known prior to loading the vehicles. He develops a variety of heuristics and tests their performance with respect to cycle time. In addition, he develops a partially discrete fluid model of the system which serves as an aid to obtaining good scheduling policies.

Dai and Weiss [10] analyze the fluid model of a MQN with ring topology and unidirectional routing. They use it to demonstrate that the network is stable under any non-idling policy when the UTCs are satisfied. The main difference between the network that they analyze and the CR is that in their model, service can begin on any job at a non-integer point of time, while in the CR, service can commence only at integer points of time. We provide more details on this paper in section 2.3.1 as it provides a crucial part of one of our

arguments.

In the context of more general fluid networks, there is a wealth of literature on finding the optimal policy with respect to various objectives such as holding cost minimization. Chen and Yao [7] define the notion of a globally optimal policy, which is a policy that minimizes the total fluid level weighted by fluid holding costs at every time t. They also show by an example that it is so strong a notion that globally optimal policies might not even exist in many cases.

Weiss has done extensive work on finding optimal policies for fluid networks. In [22] he analyzes optimal draining of re-entrant fluid lines with respect to objectives such as minimizing draining time, infinite horizon holding costs or holding costs at a target time T. A re-entrant fluid line is one in which fluid follows a fixed sequence of buffers and may visit one or more stations multiple times. He also proved that for a single station re-entrant line the last-buffer-first-served (LBFS) priority policy is globally optimal for equal holding costs. Further, in a subsequent paper [23] he analyzes the case of re-entrant lines with multiple stations. In this paper he specifically investigates the problem of minimizing average WIP over a finite time horizon.

The problem of optimizing holding costs in the fluid model falls into a general class of problems called separated continuous linear programs (SCLPs). Weiss [24], presents a new simplex-like algorithm for solving SCLPs with linear data. He then uses the fluid solution thus obtained to provide a heuristic schedule. The complete details of the simplex-like algorithm are presented in [25].

Avram et al. [3] use Pontryagin's maximum principle to solve specific holding cost problems in multiclass fluid networks. They develop a discretization method to solve the optimization problem. A learning heuristic which is numerically efficient is proposed.

1.4 Goals

In this dissertation we show that all non-idling policies are throughput optimal for some special cases of the CR. A policy is throughput optimal for a class of networks, if the associated queue length process is rate stable under the policy, whenever the UTCs are satisfied. The notion of non-idling in the CR is slightly different from the usual multiclass sense. Stability is proved by showing that the fluid model of a CR (for some cases) is identical to the fluid model of a MQN operating under a non-idling policy in the traditional sense. Definitions of stability and other requisite information are provided in section 2.3.

We also use the fluid model of these systems to find optimal policies with respect to WIP. For some simple networks we find the policy which minimizes the WIP in the queueing network and compare it with the results from the fluid model. A holding cost optimal policy is also specified for a simple example of this network. We show by an example that, with respect to WIP, an idling policy might perform better than a non-idling policy in some cases. In this context idling is equivalent to not loading a customer upon arrival of an empty vehicle to a station in which jobs are waiting in line.

1.5 Overview

In chapter 2, we describe the multiclass model of the CR and develop the corresponding queueing network equations for this model. Background on notions of stability is also provide in this chapter.

In chapter 3, we analyze the stability of a special case of the CR in which external arrivals occur to only station 1. Since any job requests less than a full rotation around the ring, this implies that no job requests a destination past station N. For this case, we show that the queueing model is rate stable (see section 2.3) under all non-idling CR policies when the UTCs are satisfied. We also show that when there is no initial population at any station except station 1, all policies have equal long run average WIP. Finally we demonstrate that the globally optimal policy for the fluid network is the SRTT policy.

Chapter 4 considers a more general case in which external arrivals occur to all stations and no job requests a destination past station N. Again, we prove that the system is rate stable under all non-idling policies when the UTCs are satisfied. We then proceed to show that the globally optimal policy for the fluid model is the SRTT policy. By means of a deterministic counterexample, we demonstrate that for the queueing network the SRTT policy is not WIP optimal. The holding cost minimization problem for the fluid model is formulated as a SCLP. The solution of this holding cost minimization problem for a simple example is presented. Note that the network analyzed in chapter 3 is a special case of the network analyzed in this chapter. Stability of the general case implies the stability of the network considered in 3. We have analyzed the

two as separate cases as it aids with the flow of the dissertation. Also some of the results for the more general case are not applicable for the special case. For example, there is no situation in which idling will be optimal with respect to the long run average WIP for the simpler version of this network.

Further in chapter 5, we prove that for a CR with N stations but only a single vehicle, idling policies are better than non-idling policies with respect to WIP. We also present simulation studies of this example in which we test the performance of some idling policies. In these simulation studies, we assume that the job interarrival times to each station are exponential. In addition, we show that idling may sometimes be better than non-idling for the CR with respect to WIP by means of another deterministic example. Finally conclusions and future research are presented in chapter 6.

Chapter 2

Modeling Framework

2.1 Detailed Model Description

In section 1.1, we briefly described the operation of the CR system and the interactions between vehicles and jobs at each station. In this chapter we begin with a detailed description of the model. Here we describe a general multiclass unidirectional ring type queueing network and show how the CR can be modeled with an operational constraint in this setting.

Consider a MQN with the N stations, numbered $1, \ldots, N$, with a ring type topology and unidirectional routing as shown in figure 2.1. We call this network the UMQN. Jobs at each station are classified into different job classes based on destination. Jobs change classes as they transit through the network. We designate a job as belonging to class (i, j), if it is at station i and it requires service at j stations along the ring before it reaches its destination. Each station may service multiple job classes, but a particular class (i, j) is only served at a single station i.

Each class can receive two types of arrivals - internal and external. An internal arrival occurs due to a job completing service from the previous station along the ring. An external arrival occurs from outside the system from an external

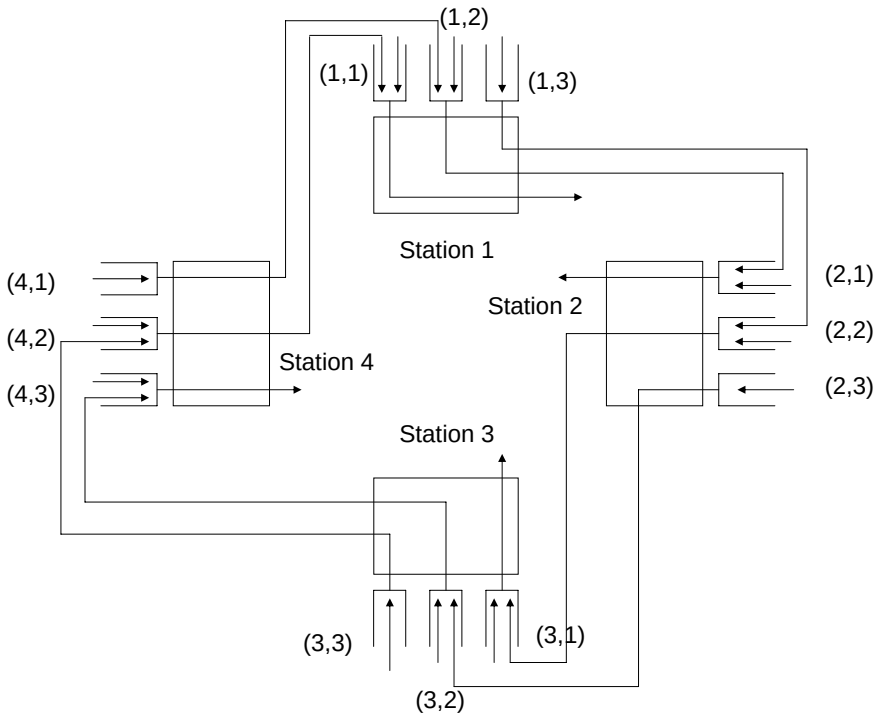

Figure 2.1: Unidirectional ring type MQN

arrival process. The details on these external arrival processes are provided later in this section.

A route is an ordered sequence of classes. Suppose a job arriving at a station i requests service at n stations along the ring as its route. The job then receives service at stations $i, i+1, \ldots, i+n-1$ if $i+n \leq N$. Note that the job leaves the system after service at station $i+n-1$. The route followed by this job would then be the sequence of classes $(i,n),(i+1,n-1),\ldots,(i+n-1,1)$. If $i+n > N$, then the job receives service at stations: $i, i+1, \ldots, N, 1, \ldots, i+N-n-1$. The route followed by this job would then be the sequence of classes

$(i, n), (i+1, n-1), \ldots, (N, N-i+1), \ldots, (i+n-N-1, 1)$.

We also assume that external arrivals to a class (j, k) are identical to internal arrivals occurring from class $(j-1, k+1)$. In proving throughput optimality, this does not matter as internal arrivals have the same service requirement as external arrivals.

We assume that each job requests a route that ends before its original station of arrival. That is, each job exits the system before it re-visits its station of arrival. Hence each station can potentially have $N-1$ job classes. Every job requests a deterministic service time of one unit from the stations along its route.

The above MQN represents a more general class of networks than the CR. Note that in the CR the inter-station travel time can be viewed as a deterministic service time at each station. Then the CR is equivalent to the UMQN with the following operational constraints:

1. Service at each station can begin only at integer points of time.

2. At each station the job to commence service at the next integer unit of time is decided by a head of the line scheduling policy (HL, see below).

Note that in the CR whenever a new arrival occurs at a station, it will have to wait for an empty vehicle to arrive at the station before service begins. This necessitates the inclusion of operational constraint (1). At the very least service cannot occur until the next integer time point. Hence the construction

of the system "enforces" idling under certain circumstances. Also note that the only jobs to actually experience a delay due to idling will be those arriving at a non-integer point of time to an empty station. Hence the CR can be viewed with a specific operating rule in the UMQN. Note that in addition to this operation a scheduling policy Π, still has to be chosen to resolve contention for service when multiple jobs are available at the station for service.

Common scheduling policies are FIFO, LRTT, SRTT, etc. LRTT and SRTT are both subsets of a class of policies known as Static Buffer Priority (SBP) policies. All of these policies are non-idling policies in the usual multiclass sense. For example under FIFO in the general ring type network, the jobs would be ordered based on arrival time. Among the jobs in the queue at each station, the job which arrived first at the station would be selected for service whenever a choice of jobs to be worked on is to be made. Under a CR FIFO policy, jobs would be ordered based on arrival time and among the jobs in all the classes at each station, the job which arrived first at the station would be selected for service at the next integer unit of time. Whenever we refer to a scheduling policy Π, in the context of the CR we refer to it as a CR-Π policy. Whenever we refer to a non-idling policy in the CR, we refer to a policy which at any station i, does not allow an empty vehicle to pass on to the next station empty when there are jobs waiting to be loaded at i.

A head of the line (HL) policy [14] is one in which a maximum of one job of each class at a given station can receive service at time t. In addition within a class (i, j), jobs are served in FIFO order. In this dissertation we only consider

HL policies. An example of a non-HL policy is Egalitarian Processor Sharing, where all jobs present receive an equal share of the station's service.

2.1.1 The Queueing Network Equations

In this section we formally define a class of queueing models with ring topology and unidirectional routing. If there exists an external arrival process to class (i, j), then let $E_{i,j}(t)$ represent the cumulative exogenous arrival process to class (i, j). This process counts the number of external arrivals up to time t. We assume that the external arrival process to class (i, j), is a function which is right continuous with left limits (RCLL) and satisfies a Strong Law of Large Numbers (SLLN) type assumption. That is, we assume that with probability 1:

$$\lim_{t\to\infty} \frac{E_{i,j}(t)}{t} = \alpha_{i,j}.$$

It is also assumed that the arrival processes are independent of each other and all other events in the system.

$\alpha_{i,j}$ are the arrival rates to class (i, j). The space of RCLL functions is denoted by $\mathbb{D}$. Let $X \in \mathbb{D}$ and Y be defined as:

$$Y(t) = \sup\{s \geq 0 : X(s) \leq t\}.$$

$Y(t)$ is known as the inverse process of $X(t)$.

Proposition 2.1.1. *(Chen and Yao [7]) Consider the (X, Y) pair. Suppose*

$$\frac{X(t)}{t} \to m \quad a.s \quad as \quad t \to \infty$$

for a positive constant m, and set $\mu = \frac{1}{m}$. Then,

$$\frac{Y(t)}{t} \to \mu \quad a.s \quad as \quad t \to \infty.$$

Furthermore as $n \to \infty$,

$$\bar{X}^n(t) \equiv \frac{X(nt)}{n} \to mt,$$

$$\bar{Y}^n(t) \equiv \frac{Y(nt)}{n} \to \mu t,$$

where, convergence is almost sure (a.s) uniformly on compact sets (u.o.c).

The proof of the above proposition can be found in Chen and Yao [7]. For each class (i, j), we let:

- $A_{i,j}(t)$ represent the total number of arrivals (internal and external) to class (i, j) in $[0, t]$
- $T_{i,j}(t)$ represent the total amount of time that station i has spent processing jobs of class (i, j) in $[0, t]$
- $D_{i,j}(t)$ represent the number of class (i, j) jobs that station i has processed in $[0, t]$
- $Q_{i,j}(t)$ represent the total number of jobs in class (i, j) at time t
- $I_i(t)$ be the total amount of time station i was idle in $[0, t]$.

Note that the idle time $I_i(t)$ includes the "enforced" idle time mentioned earlier.

The number of jobs initially in the system is denoted by the vector $Q(0) = [Q_{i,j}(0)]$. Let $T(\cdot) = [T_{i,j}(\cdot)]$, be the vector of allocation processes $\{T_{i,j}(t),\ t \geq 0\}$. Similarly let $E(\cdot) = [E_{i,j}(\cdot)]$, be the vector of external arrival processes $\{E_{i,j}(t),\ t \geq 0\}$. The indices i and j run from $\{1, \ldots, N\}$ and $\{1, \ldots, N-1\}$ respectively. Given these processes, we write down the set of dynamical equations known as the queueing network equations below:

$$
\begin{aligned}
A_{i,j}(t) &= E_{i,j}(t) + D_{i-1,j+1}(t) && (2.1.1)\\
Q_{i,j}(t) &= Q_{i,j}(0) + A_{i,j}(t) - D_{i,j}(t), && (2.1.2)\\
D_{i,j}(t) &= \lfloor T_{i,j}(t) \rfloor, && (2.1.3)\\
D_{0,j}(t) &= D_{N,j}(t) && (2.1.4)\\
W_i(t) &= \sum_{j=1}^{N-1} Q_{i,j}(t) && (2.1.5)\\
I_i(t) &= t - \sum_{j=1}^{N-1} T_{i,j}(t), && (2.1.6)\\
T_{i,j}(0) &= 0 \text{ and } T_{i,j}(\cdot), I_i(\cdot) \text{ are non-decreasing}, && (2.1.7)\\
Q_{i,j}(t) &\geq 0. && (2.1.8)
\end{aligned}
$$

Note that we set the departure process at the N^{th} station to be equal to $D_{0,j}(t)$. This is simply a logical construct necessitated by the circular topology of the network. The equations (2.1.2) represent the balance constraints. In addition to (2.1.1)-(2.1.8), we need to specify the operating policy as well. This imposes an additional constraint on the allocation process.

Non-Idling Condition:

A non-idling operation of the CR imposes the constraint that the idle time of any station i, $I_i(t)$ cannot increase unless there are no jobs at the station or if an arrival occurs to an empty station i at a non-integer point of time. This condition is:

at each station i, for all $t_2 > t_1 \geq 0, I_i(t_2) - I_i(t_1) > 0$ only if,

$$t_1 \in Z^+, t_1 < t_2 \leq t_1 + 1, \exists\, s:\ t_1 < t_2 \leq t_1 + 1 \text{ and } W_i(s) = 0 \text{ or}$$
$$W_i(t) = 0,\ \forall\, t\ \in [t_1, t_2).$$

For the UMQN without the operational restrictions of the CR, the usual non-idling condition is simply (2.1.9a).

To complete the specification of the queueing network equations we also need equations to enforce the scheduling policy Π. Since we intend to show stability of the CR under all non-idling policies we do not include an equation to specify the policy.

2.1.2 Throughput and Traffic Intensities

In this section we define the effective arrival rates and traffic intensities for the UMQN. The traffic intensity is then expressed in terms of the effective arrival rate of the various classes. The usual traffic conditions for the UMQN are also defined. The definitions in this section lay the groundwork for the stability conditions for the UMQN.

Definition 2.1.1. The *effective arrival rate* for class (i, j) is the total arrival rate to class (i, j) calculated by considering external arrivals and internal

arrivals. The effective arrival rate for class (i,j) is defined as:

$$\lambda_{i,j} \equiv \alpha_{i,j} + \sum_{k=1}^{i-1} \alpha_{i-k,j+k} + \sum_{k=i+1}^{N-1} \alpha_{N+i+1-k,k}. \tag{2.1.10}$$

The nominal load per unit of time at station i is known as the traffic intensity. Let ρ_i represent the traffic intensity at station i. Since the service rate at each station for each class is one unit of time, the total arrival rate to a station is equal to the nominal load per unit of time presented to the station. It was previously widely believed that multiclass networks which satisfied the condition that the traffic intensity at each station is less than one were stable under all non-idling policies. This notion has now been disproved by a number of examples (see [18, 21]). The conditions that the nominal load per unit time is less than one at each station are known as the usual traffic conditions (UTCs). The UTCs for the CR are listed below. For each $i \in \{1, \ldots, N\}$:

$$\rho_i \equiv \sum_{j=1}^{N-1} \lambda_{i,j} \leq 1. \tag{2.1.11}$$

For the system shown in figure 2.1, the traffic intensity at station 1 is:

$$\rho_1 = \alpha_{1,1} + \alpha_{1,2} + \alpha_{1,3} + \alpha_{4,2} + \alpha_{4,3} + \alpha_{3,3}.$$

Definition 2.1.2. For a class (i,j), if there exists a constant $\tilde{\lambda}_{i,j}$ such that with probability 1:

$$\lim_{t\to\infty} \frac{D_{i,j}(t)}{t} = \tilde{\lambda}_{i,j},$$

then $\tilde{\lambda}_{i,j}$ is known as the *throughput* of class (i,j).

2.2 The Fluid Model

The fluid model is one of the most commonly used and effective techniques to analyze stability of a queueing network. The stability of a fluid model (section 2.3) is relatively easy to establish. The connection between the stability of the fluid model and that of the queueing network has been analyzed by Dai [8], Chen [6] and others. The fluid model is obtained by replacing the stochastic processes in the queueing model by their continuous deterministic analogs. The jobs in the system are no longer viewed as discrete entities. We regard them as flowing continuously through the network and hence the term fluid model.

Let $Q(\cdot)$ represent the vector of queue lengths for the UMQN. As n goes to infinity, the scaled process $Q(nt)/n$ converges to a deterministic process satisfying the fluid model equations (2.2.1) to (2.2.8). This type of scaling in time and space is referred to as fluid scaling. The scaling factor n may represent the initial number of jobs in the network or some other system parameter. For example in Coffman et al. [12], the scaling factor is chosen as the number of vehicles. A superscript n is used to indicate the dependence of the process on the scaling factor n. For example, the fluid scaling of the process $Q(t)$ is given by:

$$\bar{Q}^n(t) = \frac{Q^n(nt)}{n}.$$

$(\bar{Q}(t), \bar{T}(t))$ is a fluid limit of the joint process $(Q(t), T(t))$ if for some sample

path ω and a sequence $n_k \to \infty$,

$$(\bar{Q}^{n_k}(t,\omega), \bar{T}^{n_k}(t,\omega)) \to (\bar{Q}(t), \bar{T}(t)) \text{ u.o.c.}$$

Chen and Yao [7] show that if the fluid limit exists, it satisfies the fluid model equations (2.2.1) to (2.2.8). It is to be noted that every fluid limit is a solution to the fluid model equations but not vice versa [9].

The *fluid model* is the set of all solutions to the fluid model equations (2.2.1) to (2.2.8). The fluid quantities are represented with a bar. For all $i \in \{1, \ldots, N\}$, $\forall\, j \in \{1, \ldots, N-1\}$, and $t \geq 0$, the HL fluid model equations for the UMQN are:

$$\bar{A}_{i,j}(t) = \alpha_{i,j} t + \bar{D}_{i-1,j+1}(t) \tag{2.2.1}$$

$$\bar{Q}_{i,j}(t) = \bar{Q}_{i,j}(0) + \bar{A}_{i,j}(t) - \bar{D}_{i,j}(t), \tag{2.2.2}$$

$$\bar{D}_{i,j}(t) = \bar{T}_{i,j}(t), \tag{2.2.3}$$

$$\bar{D}_{0,j}(t) = \bar{D}_{N,j}(t) \tag{2.2.4}$$

$$\bar{W}_i(t) = \sum_{j=1}^{N-1} \bar{Q}_{i,j}(t) \tag{2.2.5}$$

$$\bar{I}_i(t) = t - \sum_{j=1}^{N-1} \bar{T}_{i,j}(t), \tag{2.2.6}$$

$$\bar{T}_{i,j}(0) = 0 \text{ and } \bar{T}_{i,j}(\cdot),\ \bar{I}_i(\cdot) \text{ are non-decreasing}, \tag{2.2.7}$$

$$\bar{Q}_{i,j}(t) \geq 0. \tag{2.2.8}$$

The fluid model equation for the non-idling condition is:

At each station i, $\bar{I}_i(t_2) - \bar{I}_i(t_1) > 0$ for all $t_2 > t_1 \geq 0$, if and only if:

$$\bar{W}_i(t) = 0,\ \forall\, t \in [t_1, t_2). \tag{2.2.9}$$

Definition 2.2.1. The set of all feasible solutions to the fluid model equations (2.2.1) to (2.2.9) is referred to as the *non-idling fluid model* of the UMQN.

Note that

$$\lim_{n\to\infty} \frac{E^n_{i,j}(nt)}{n} = \alpha_{i,j}t.$$

This follows from the assumption about the external arrival process and proposition 2.1.1. The fluid model equations (2.2.1-2.2.9) follow directly from (2.1.1-2.1.9a). The fluid model equations above are a special case of those developed by Dai and Weiss ([10]) for the UMQN. In the fluid model $\bar{Q}_{i,j}(t)$ is interpreted as the total amount of fluid present in buffer (i,j) at time t. $\bar{D}_{i,j}(t)$ represents the total quantity of type (i,j) fluid that was processed by station i during $[0,t]$. $\bar{T}_{i,j}(t)$ is the total amount of time spent by station i working on fluid of type (i,j) during $[0,t]$. As mentioned in section 2.1.1, additional constraints need to be imposed in order to enforce the specific scheduling policy employed. Equation (2.2.2) is the flow balance constraint. The total amount of fluid input to class (i,j) up to time t consists of the class (i,j) fluid present initially, the total amount of fluid which arrived externally to class (i,j) and the total amount of fluid processed at station $i-1$ of class $(i-1,j+1)$ in $[0,t]$. The total quantity of class (i,j) fluid processed in $[0,t]$ is given by $\bar{D}_{i,j}(t)$. Hence the amount of fluid in class (i,j) at time t is the difference between the input and the output up to time t. Equations (2.2.6) and (2.2.7) together imply that a station cannot spend more than 100% of its time processing fluid in that station.

2.3 Stability

There are various notions of stability associated with queueing networks and their fluid models. In this section we define the notion of stability for both queueing networks and their fluid models and explain the relation between them. Note that we only define the notions of stability we plan to use in this dissertation. Finally we discuss the stability of the ring type MQN without the operational restrictions of the CR.

Definition 2.3.1. The UMQN is *rate stable* if for every fixed initial data, with probability 1,

$$\lim_{t\to\infty} \frac{D_{i,j}(t)}{t} = \lambda_{i,j} \quad \forall\, i \in \{1,\ldots,N\} \text{ and } \forall\, j \in \{1,\ldots,N-1\},$$

where $\lambda_{i,j}$ is the effective arrival rate for class (i,j) defined in 2.1.1.

Definition 2.3.2. A scheduling policy is called *throughput optimal* for the CR if it is rate stable under the usual traffic conditions, defined by (2.1.11).

Definition 2.3.3. The fluid model is *weakly stable* if for each fluid solution such that $|Q(0)| = 0$, $Q(t)$ is zero for all $t \geq 0$.

The only tasks that remain to be done in this chapter are to connect the stability of the fluid model with that of a corresponding queueing network and to establish the stability of the fluid model for the ring type MQN. The following theorem states that the weak stability of the fluid model implies that the corresponding queueing network is stable. The result holds for networks operating under HL policies.

Theorem 2.3.1. *(Chen [6]) If the fluid model is weakly stable, then any corresponding queueing network is rate stable.*

Apart from this, there exist various notions of stability in both fluid and stochastic networks. There are also different notions of instability defined for both networks. We refer the reader to Dai [9] for further details.

2.3.1 Stability of the UMQN

Dai and Weiss [10] discuss the stability of the UMQN. In that paper it was shown that the fluid model of this ring type MQN network is stable under the UTCs under all non-idling policies. That is, if the traffic intensity, $\rho_i < 1$ at each station i, the system is stable under any non-idling policy. Dai and Weiss accomplish this using a Lyapunov function approach [9]. With a minor modification in their proof, the following result can be established.

Theorem 2.3.2. *The fluid model of the UMQN is weakly stable under all non-idling policies if* $\rho_i \leq 1$.

Hence by theorem 2.3.1, any corresponding queueing network is rate stable. Note that stability is proved only for the ring type MQN without the operational constraint of the CR imposed on it.

Chapter 3

Simple Feed-forward Cambridge Ring

In this chapter we describe and analyze the simplest version of the CR that we consider. This network is a special case of the UMQN described in section 2.1. We assume that any external arrival to the system occurs only at station 1. Jobs arriving to station 1 request a particular destination station along the ring. Jobs with different destinations are divided into different job classes. Since every job requests a destination which is less than one full rotation around the ring, an empty vehicle reaches station 1 every unit of time. As in the previous chapter we describe below how this system can be viewed under the MQN setting with an additional operational constraint. This model is known as the *Simple Feed-forward Cambridge Ring (SFCR)*.

3.1 Detailed Model Description

Consider the UMQN with N stations. All external arrivals occur at station 1. The assumptions made on the external arrival processes are as described in subsection 2.1.1. Jobs arriving to station 1 request service at a sequence of stations along the ring. At each station every job requests a service of one unit of time. As mentioned in section 2.1, this deterministic service time at

each station is equivalent to the interstation travel time. Jobs are grouped into classes based on the number of stations at which service is requested. At station 1, the arriving jobs are thus grouped into one of potentially $N-1$ classes. The job changes classes as it transits through the network. Any class at a station i (> 1) receives only internal arrivals. A job of class (i, j) is currently at station i with j steps remaining to reach its destination station.

The CR restrictions force service to begin at every station at only integer points of time. Since in this problem no job requests a destination beyond station N, station 1 is free to start service on a new job at every integer point of time. The ring type MQN with unidirectional routing under these operational restrictions is equivalent to the SFCR. Note that these restrictions occasionally "enforce" idling in the system as mentioned in section 2.1.

We also need a policy to resolve contention for service when there are jobs of different classes present at any station i. We call this scheduling policy Π. The MQN operating under the CR restrictions and a scheduling policy Π is referred to as operating under the SFCR-Π policy. We assume that the SFCR-Π policy is a HL scheduling policy. For example if we were using FIFO for the SFCR, the jobs in the queue at each station would be sorted according to arrival times and service would begin on the job with the earliest arrival time at the next integer point of time. This policy is referred to as the SFCR-FIFO policy. An SFCR with three stations is shown in figure 3.1.

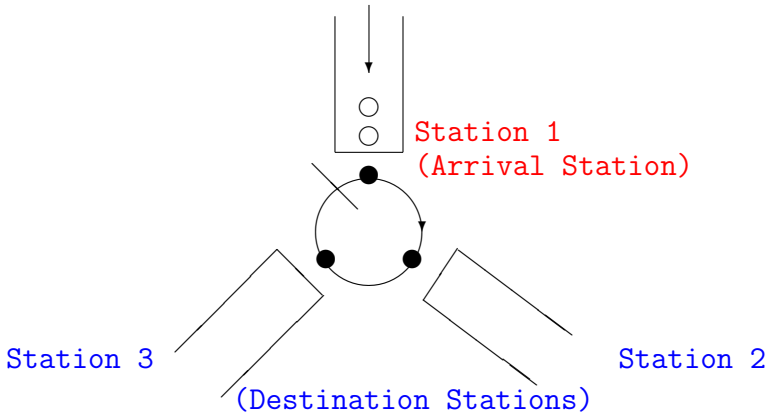

Figure 3.1: Simple Feed-forward Cambridge Ring

3.1.1 Queueing Model Equations

In this section we develop the queueing model equations for the MQN operating under the SFCR-Π policy, where Π is an arbitrary non-idling HL policy. We use a notation similar to 2.1 to refer to the queueing model quantities. Since we intend to compare the queueing model quantities along every sample path between two different policies we introduce a superscript notation. The superscript C refers to the queueing model quantities when the UMQN operates under the SFCR-Π policy. For example, we let $A^C_{i,j}(t)$ be the total number of arrivals (external and internal)to class (i,j) in $(0,t]$ under the SFCR-Π policy.

The quantities $A^C_{i,j}(t)$, $T^C_{i,j}(t)$, $D^C_{i,j}(t)$ and $Q^C_{i,j}(t)$ are compared along an arbitrary sample path of external arrivals ω, to the corresponding quantities when the system operates under an MQN policy described in section 3.2.2. Hence we require this superscript to differentiate between the two systems. Since the external arrivals are not affected by the choice of scheduling policy, $E_{i,j}(t)$ still represents the exogenous arrival process to class (i,j).

The queueing network equations under the SFCR-Π policy are defined by (2.1.1)-(2.1.8) with a few modifications. The additional restrictions needed are:

$$D^C_{0,j}(0) = 0 \tag{3.1.1}$$

$$E_{i,j}(t) = 0 \ \forall \, i > 1. \tag{3.1.2}$$

Equation (3.1.1) represents the condition that no job requests a destination past station N, while (3.1.2) enforces the fact that external arrivals occur only to station 1. The non-idling condition is specified by (2.1.9a) and (2.1.9a) as before. An operational constraint is that service can only begin at integer units of time at all stations. Additional equations are necessary to enforce the scheduling policy Π.

3.2 Stability Analysis

In this section we intend to show that all non-idling policies Π are throughput optimal for the UMQN operating under the SFCR restrictions. We accomplish this by showing the rate stability of this system for all non-idling policies when the UTCs are satisfied.

Theorem 3.2.1. *In the SFCR all non-idling scheduling policies are throughput optimal.*

Consider the UMQN with N stations operating under an arbitrary but fixed SFCR-Π policy as described in section 3.1. Our approach to proving 3.2.1

involves the use of fluid models. However we do not directly construct a fluid model for this network. We describe a scheduling policy for the MQN, referred to as the MQN-Π policy, which closely resembles the SFCR-Π policy except for the fact that the "enforced" idling is eliminated. It is then shown that the fluid limit of the MQN under the SFCR-Π policy is identical to that of the MQN operating under the MQN-Π policy. The remainder of this section outlines the intermediate steps in proving this theorem.

3.2.1 Usual traffic conditions

Let $\alpha_{1,j} (\geq 0)$ be the arrival rate to class $(1, j)$. The UTCs for this system are:

$$\rho_i \equiv \sum_{j=1}^{N-i} \alpha_{1,j} \leq 1, \; \forall \, i \; \in \; \{1, \ldots, N\}.$$

In this case it is obvious that if the UTC at station 1 is satisfied, the UTCs at all other stations are also satisfied. Hence station 1 determines the stability of the system.

The MQN version of a three station SFCR is shown in figure 3.2. Note that the last station in this problem serves only as a destination and does not transport jobs forward. There is no service requested by any job at this station.

3.2.2 The MQN-Π Policy

Below we describe a policy which eliminates idling from the SFCR-Π policy and yet maintains the order of arrival to the next station. We accomplish this

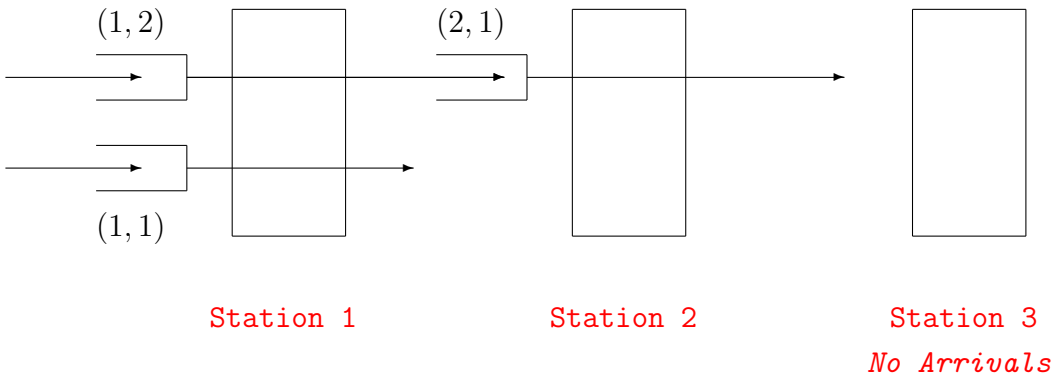

Figure 3.2: Multiclass Version of SFCR

by letting the choice of job under this non-idling policy be controlled by the SFCR-Π policy at every integer point of time. We refer to this policy as the MQN-Π policy. A busy period at station 1 begins when an arrival occurs to an empty station. Suppose that we have defined a fixed, but arbitrary non-idling (in the CR sense) policy SFCR-Π. Then the corresponding MQN-Π policy is defined as follows:

Under this policy at station 1:

1. If a job is to be chosen for service at any non-integer time, the choice is made based on the scheduling policy Π applied to the jobs waiting in the queues. Service commences on the chosen job immediately and is continued until the next integer point of time.

2. At the next integer point of time, one of the following events may occur:

   - No other exogenous arrival may have occurred at station 1 until

this time or other jobs have arrived at station 1 which, under the SFCR-Π policy have lower priority than job in service. In this case service continues on the job currently in service. Upon completion of service repeat step 1.

- Other jobs have arrived at station 1 which, under the SFCR-Π policy have higher priority. In this case, service on the current job is pre-empted and the job with highest priority under the SFCR-Π policy is chosen for service. Service is resumed on the pre-empted job at some integer point of time before the end of the busy period in the MQN-Π policy. Upon completion of service on this job repeat step 1.

Remark 3.2.1. The MQN-Π policy is set up so that under both policies the system would have the same choice of jobs at all integer points of time. Any non-idling SFCR-Π policy can be translated to a corresponding MQN-Π policy.

The superscript M is used to represent quantities in the queueing network equations under the MQN-Π policy. For example, $T^M_{1,k}(t)$ is the total amount of time spent processing jobs of class $(1,k)$ in $[0,t]$ under the MQN-Π policy. The queueing network equations under the MQN-Π policy are similar to the queueing network equations (2.1.1)-(2.1.8) and (3.1.1). All quantities in these equations have the superscript M. The restriction that service begins upon the integer unit of time is dropped. Constraints to enforce the arbitrary but fixed non-idling policy Π would still be needed. Note that since the MQN-Π

policy is a non-idling policy, we need a condition to enforce non-idling. This condition is simply (2.1.9a).

Lemma 3.2.2. *For a fixed but arbitrary non-idling policy* Π *at station* 1, $0 \leq T^M_{1,j}(t,\omega) - T^C_{1,j}(t,\omega) \leq 1, \forall\, j \in \{1,\ldots,N-1\}$, $t \geq 0$ *and any sample path* ω.

Remark 3.2.2. As in subsection 3.1.1 we refer to a sample path of external arrivals. Note that the processes $T^M_{i,j}(t)$ and $T^C_{i,j}(t)$ depend on ω. Every quantity involved in the queueing network equations depends on the sample path ω. Henceforth in order to simplify notation, the omega will be suppressed in the proofs that follow.

Proof. Fix a sample path ω. Let τ_B be the first non-integer time at which an arrival occurs to an empty station 1. Note that τ_B is also sample path dependent. Until time τ_B, $T^M_{1,j}(t) - T^C_{1,j}(t)$ is zero for all classes $(1,j)$. Similarly we define δ_B as the earliest time after τ_B, when under the SFCR-Π policy the station is empty again. We would like to analyze the difference between the quantities, $T^M_{1,j}(t)$ and $T^C_{1,j}(t)$, for every class over the duration $(\tau_B, \delta_B]$. The reason that we choose this period is that this is the first interval in which under at least one of the policies (MQN-Π policy or the SFCR-Π policy), the system is busy. Note that there can be multiple busy periods under the MQN-Π policy within $(\tau_B, \delta_B]$.

Suppose this arrival belongs to class $(1,k)$. We label this arrival n_k. For $t \in [\tau_B, \lceil \tau_B \rceil]$, $T^M_{1,k}(t)$ increases from 0 to $\lceil \tau_B \rceil - \tau_B$ as under the MQN-Π policy, the

network commences service on this job immediately. However $T^C_{1,k}(t)$ remains at zero as under the SFCR-Π policy the network does not start working on this job until $\lceil \tau_B \rceil$.

Two cases arise:

1. Between time τ_B and $\lceil \tau_B \rceil$, jobs arrive to one or more classes which have higher priority than class $(1, k)$ under the SFCR-Π policy. Again let us label the highest priority job under this policy at time $\lceil \tau_B \rceil$ as m_l. Under the SFCR-Π policy, the system starts operating on job m_l at time $\lceil \tau_B \rceil$. Under the MQN-Π policy, job n_k receives $\lceil \tau_B \rceil - \tau_B$ units of service. Job n_k is pre-empted by job m_l at time $\lceil \tau_B \rceil$.

2. In the interval $(\tau_B, \lceil \tau_B \rceil]$, any jobs that arrive to the system belong to classes that have lower priority than the class $(1, k)$ job n_k under the SFCR-Π policy. Hence under the SFCR-Π policy, the UMQN starts operating on job n_k at the next integer unit of time. Under the MQN-Π policy job n_k, has already been served for $\lceil \tau_B \rceil - \tau_B$ units of time. Service simply continues on job n_k at time $\lceil \tau_B \rceil$.

Case 1:

Suppose then that n_k is preempted at $\lceil \tau_B \rceil$. At time $\lceil \tau_B \rceil$, $T^M_{1,k}(\tau_B) - T^C_{1,k}(\tau_B) = \lceil \tau_B \rceil - \tau_B \, (\leq 1)$. This difference remains constant until job n_k resumes service. Note that under the MQN-Π policy, job n_k will resume service only at an integer point of time before the current busy period ends.

At time $\lceil \tau_B \rceil$, service begins on the job with highest priority: m_l. At this point under both the MQN-Π and the SFCR-Π policies, the UMQN start processing jobs in the same order. Hence for all other classes, $(1, k')$, $k' \in \{1, \ldots, N-1\} \backslash \{k\}$ that are processed until job n_k resumes service, $T^M_{1,k'}(t) - T^C_{1,k'}(t) = 0$.

Under the MQN-Π policy, before the busy period ends, the preempted job of class $(1, k)$ will have to be processed. Under the MQN-Π policy the amount of service remaining on this job is $1 - (\lceil \tau_B \rceil - \tau_B)$ units of time, while under the SFCR-Π policy it is 1 unit of time.

Under the MQN-Π policy the system begins processing this job at some integer point of time in the future. Let this time be t_R. This will occur when the class $(1, k)$ job, n_k, becomes the highest priority job under the SFCR-Π policy. When service on job n_k resumes, for $t \in [t_R, t_R+1-\lceil \tau_B \rceil+\tau_B]$, $T^M_{1,k}(t)-T^C_{1,k}(t) = \lceil \tau_B \rceil - \tau_B$.

At time $t_R+1-\lceil \tau_B \rceil+\tau_B$, if there are any jobs remaining in any class, under the MQN-Π policy the system starts processing a job of the class with the highest priority under the policy Π at this time. Under the SFCR-Π policy the system continues processing job n_k, until $t_R + 1$. At $t_R + 1 - \lceil \tau_B \rceil + \tau_B$, if under the MQN-$\Pi$ policy the system starts service on a job of a class $(1, k')$, $(k' \neq k)$, $T^M_{1,k}(t) - T^C_{1,k}(t)$ decreases to zero at time $t_R + 1$. However if at time $t_R+1-\lceil \tau_B \rceil-\tau_B$, under the MQN-$\Pi$ policy the system starts processing a new job of class $(1, k)$, the difference remains constant. If under the MQN-Π policy, service begins on a job of any other class, say $(1, k)'$ at time $t_R+1-\lceil \tau_B \rceil-\tau_B$ at time $t_R + 1$, $T^M_{1,k'}(t_R + 1) - T^C_{1,k'}(t_R + 1)$ is equal to $\lceil \tau_B \rceil - \tau_B$.

Thus the differences get transferred from class to class until the end of a busy period in the MQN-Π policy. However there can only be a maximum of two classes with a positive difference at the same instant of time. One of the situations that needs to be examined in this case is what happens when there are multiple MQN-Π policy busy periods within a single SFCR-Π policy busy period.

At the end of a busy period under the MQN-Π policy, let class $(1, l)$ be the last class processed. Let t_L be the time at which this job concludes service under the MQN-Π policy. Under the SFCR-Π policy service concludes on this job at time $\lceil t_L \rceil$. If there are no arrivals in $[t_L, \lceil t_L \rceil)$ the differences in times served reduce to zero at $\lceil t_L \rceil$ and $\delta_B = \lceil t_L \rceil$.

If an arrival does occur in the interval $[t_L, \lceil t_L \rceil)$, then a new busy period under the MQN-Π policy begins before the ending of a busy period under the SFCR-Π policy. In this situation the system under the MQN-Π and the SFCR-Π policies would be working on jobs belonging to different classes at the same time from the time of this arrival up to time $\lceil t_L \rceil$. The system under the SFCR-Π policy would be working on class l. Let us assume that the system under the MQN-Π policy works on a job of class m'. Then $T^M_{1,l}(t) - T^C_{1,l}(t)$ and $T^M_{1,m'}(t) - T^C_{1,m'}(t)$ will be positive in the interval $(t_L, \lceil t_L \rceil)$.

Let t_S be the time of completion of the class l job under the MQN-Π policy and t_A be the time of arrival of the next job which belongs to class $(1, m')$. We assume that t_A is less than $\lceil t_S \rceil$. Otherwise the busy period in the SFCR-Π policy ends before the arrival of this job.

At time t_S:

$$T^M_{1,l}(t_S) - T^C_{1,l}(t_S) = \lceil t_S \rceil - t_S.$$

At time t_A:

$$T^M_{1,l}(t_A) - T^C_{1,l}(t_A) = \lceil t_S \rceil - t_A$$

and

$$T^M_{1,m'}(t_A) - T^C_{1,m'}(t_A) = 0.$$

For all $t \in [t_A, \lceil t_S \rceil)$:

$$T^M_{1,l}(t) - T^C_{1,l}(t) = \lceil t_S \rceil - t$$

and,

$$T^M_{1,m'}(t) - T^C_{1,m'}(t) = t - t_A.$$

The total difference, $T^M_{1,l}(t_S) - T^C_{1,l}(t_S) + T^M_{1,m'}(t_S) - T^C_{1,m'}(t_S) = \lceil t_S \rceil - t_A$ which is less than or equal to 1. At the next integer unit of time $\lceil t_A \rceil$, $T^M_{1,l}(\lceil t_A \rceil) - T^C_{1,l}(\lceil t_A \rceil)$ decreases to 0 and

$$T^M_{1,m'}(\lceil t_A \rceil) - T^C_{1,m'}(\lceil t_A \rceil) = \lceil t_S \rceil - t_A,$$

which is less than the original difference, $\lceil t_S \rceil - t_S$.

Hence under the MQN-Π policy, whenever a busy period ends the total difference between all the classes decreases. If there is no arrival before the next integer point of time then under the SFCR-Π policy, the busy period ends. At this point of time both systems have spent the same time working on all classes. Thus in this case for all $t \in [\tau_B, \delta_B]$, we have $T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1$.

Case 2:

Suppose now that under the MQN-Π policy, job n_k does not get preempted at time τ_B. At time $\lceil \tau_B \rceil$, under the SFCR-Π policy service commences on job n_k. By this time under the MQN-Π policy, $\lceil \tau_B \rceil - \tau_B$ units of service have been completed on this job. Job n_k completes service under the MQN-Π policy at time $\tau_B + 1$. At that point of time, under the SFCR-Π policy $1 - (\lceil \tau_B \rceil - \tau_B)$ units of service have been completed on job n_k. Thus at time $\tau_B + 1$, we have, $T^M_{1,k}(t) - T^C_{1,k}(t) = \lceil \tau_B \rceil - \tau_B$.

If there are no other jobs waiting in the system at time $\tau_B + 1$, this difference decreases until time $\lceil \tau_B \rceil + 1$. If there is no other arrival until time $\lceil \tau_B \rceil + 1$, $T^M_{1,k}(t) - T^C_{1,k}(t)$ decreases to zero. If there is an arrival before the next integer unit of time, it results in a situation similar to the end of busy period situation explained in case (1).

If there are jobs waiting at time $\tau_B + 1$, the MQN-Π policy picks the highest priority job under the SFCR-Π policy from among those waiting and starts service on that job. Under the SFCR-Π policy the system is currently serving job n_k. Suppose that under the MQN-Π policy the system starts service on job m_l of class $(1, l)$ at time $\tau_B + 1$. Then $T^M_{1,k}(\lceil \tau_B \rceil + 1) - T^C_{1,k}(\lceil \tau_B \rceil + 1) = 0$ and $T^M_{1,l}(\lceil \tau_B \rceil + 1) - T^C_{1,l}(\lceil \tau_B \rceil + 1) = \lceil \tau_B \rceil - \tau_B$. At this time if there are any higher priority jobs than m_l at station 1, under both the MQN-Π policy and the SFCR-Π policy the system starts working on that job. Otherwise under the SFCR-Π policy, service commences on job m_l, while under the MQN-Π policy, service simply continues. Note that under the SFCR-Π policy the system never commences service on job m_l until it becomes the highest priority job. Thus

the differences get transferred from class to class until the end of a busy period in the MQN-Π policy.

The end of the busy period scenario is exactly the same as in the previous case. Since the initial difference is less than one, and the difference can only decrease until the end of a busy period under the SFCR-Π policy (which occurs at time δ_B), when the SFCR-Π policy catches up, the difference is always less than or equal to 1 over a busy period under the SFCR-Π policy. Hence in this case as well we have for all $t \in [\tau_B, \delta_B]$, $T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1$.

Case Summary:

It is to be observed that under the SFCR-Π policy whenever a busy period ends, under the MQN-Π policy the system is idle. Hence under the SFCR-Π policy at the end of a busy period, the amount of time that each class has been served at station 1 is identical to the corresponding quantity under the MQN-Π policy. We have seen that over $(\tau_B, \delta_B]$, the difference between the total amount of time spent by each of these policies on any class $(1, k)$ is less than or equal to 1 unit of time. Whenever under the MQN-Π policy the system is idle, under the SFCR-Π policy the system is either idle or is catching up on the MQN-Π policy in terms of total service time to each class at station 1. Hence $\forall\, t > 0$, $0 \leq T^M_{1,k}(t) - T^C_{1,k}(t) \leq 1$ for all $k \in 1, \ldots, N-1$. □

Consider the following example: an arrival to class $(1, 1)$ occurs at time 0.3 and an arrival to class $(1, 2)$ occurs at time 1.4. Another arrival to class $(1, 3)$ occurs at time 1.7. Let us assume that under the SFCR-Π policy, class $(1, 3)$

takes precedence over class $(1,2)$. The table 3.1 shows the time differences in this situation.

Arrival Number	Class	Time(t)	$T^M_{1,1}(t) - T^C_{1,1}(t)$	$T^M_{1,2}(t) - T^C_{1,2}(t)$
1	(1,1)	0.3	0	0
		1.0	0.7	0
		1.3	0.7	0
2	(1,2)	1.4	0.6	0
		1.5	0.5	0.1
3	(1,3)	1.7	0.3	0.3
		2.0	0	0.6
		3.0	0	0.6
		3.4	0	0.6
		4.0	0	0

Table 3.1: Multiple MQN-Π busy periods in a single SFCR busy period at Station 1

As the example shows there are instances of time in this case when there are two classes with a positive difference between the two systems. This happens when there are multiple MQN-Π busy periods within a single SFCR-Π busy period.

Lemma 3.2.3. *For a fixed but arbitrary non-idling policy* Π, $0 \leq T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1$ *for all stations,* $i \in \{1, \ldots, N-1\}$, $j \in \{1, \ldots, N-i\}$, $\forall\, t \geq 0$ *and any sample path* ω.

Proof. This result for station 1 has already been proven in lemma (3.2.2). Hence it is enough to show this result for all downstream stations $i \in$

$\{2, \ldots, N-1\}$. We begin this proof by observing that $A^M_{i,j}(t) - A^C_{i,j}(t) \leq 1$, since,

$$\begin{aligned} A^M_{i,j}(t) - A^C_{i,j}(t) &= D^M_{i-1,j+1}(t) - D^C_{i-1,j+1}(t) \\ &= \lfloor T^M_{i-1,j+1}(t) \rfloor - \lfloor T^C_{i-1,j+1}(t) \rfloor \\ &\leq 1. \end{aligned}$$

Also note that no pre-emption occurs in the downstream stations. At any downstream station, all arrivals are internal arrivals. Further, any arrival that occurs at time t at a downstream station i under the MQN-Π policy will occur under the SFCR-Π policy at time $\lceil t \rceil$. This follows from the way the MQN-Π policy is setup at the previous station. The MQN-Π policy commences service as soon as an arrival occurs at a downstream station. This enables the SFCR-Π policy to commence service on this job at time $\lceil t \rceil$ and under the MQN-Π policy, service on this job is simply continued (since there is no other external arrival). Since there can only be a maximum of one internal arrival in the time interval $(\lfloor t \rfloor, \lceil t \rceil]$ under either policy, the total time spent serving each class at any downstream station differs by less than one integer unit of time. That is,

$$0 \leq T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1, \; \forall \, t \geq 0.$$

□

3.2.3 Equivalence of Fluid Limits Under MQN-Π and SFCR-Π Policies

For the SFCR, the initial station acts as a synchronizing station. The way the MQN-Π policy is setup ensures that $Q^C_{i,j}(t) - Q^M_{i,j}(t)$ is always non-negative.

Further, at station 1, $\forall\, t \geq 0$:

$$\begin{aligned} 0 \leq \lim_{n\to\infty} \frac{T^M_{i,j}(nt)}{n} - \lim_{n\to\infty} \frac{T^C_{i,j}(nt)}{n} &\leq \lim_{n\to\infty} \frac{1}{n} \\ &= 0. \end{aligned}$$

Hence $\forall\, t \geq 0$,

$$\bar{T}^C_{i,j}(t) = \bar{T}^M_{i,j}(t),$$

where $\bar{T}^C_{i,j}(t)$ and $\bar{T}^M_{i,j}(t)$ are the fluid limits of the allocation processes.

$$D^M_{i,j}(t) - D^C_{i,j}(t) = \lfloor T^M_{i,j}(t) \rfloor - \lfloor T^C_{i,j}(t) \rfloor,\ \forall\, t \geq 0.$$

This implies that $D^M_{i,j}(t) - D^C_{i,j}(t)$ is either zero or one, since $T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1$. It follows that $\forall\, t \geq 0$, $\bar{D}^C_{i,j}(t) = \bar{D}^M_{i,j}(t)$.

Note that $\forall\, t \geq 0$,

$$\begin{aligned} Q^C_{i,j}(t) - Q^M_{i,j}(t) &= Q^C_{i,j}(0) - Q^M_{i,j}(0) + A^C_{i,j}(t) - A^C_{i,j}(t) - D^C_{i,j}(t) + D^M_{i,j}(t) \\ &= D^M_{i,j}(t) - D^C_{i,j}(t) \\ &\leq 1. \end{aligned}$$

Therefore, $Q^C_{i,j}(nt) - Q^M_{i,j}(nt) \leq 1$, $\forall\, n,\ t \geq 0$. This implies that, under fluid scaling:

$$\begin{aligned} 0 \leq \lim_{n\to\infty} \frac{Q^C_{i,j}(nt)}{n} - \lim_{n\to\infty} \frac{Q^M_{i,j}(nt)}{n} &\leq \lim_{n\to\infty} \frac{1}{n} \\ &= 0. \end{aligned}$$

Therefore $\forall\, t \geq 0$,

$$\bar{Q}^M_{i,j}(t) = \bar{Q}^C_{i,j}(t),$$

where $\bar{Q}^C_{i,j}(t)$ and $\bar{Q}^M_{i,j}(t)$ are the fluid limits of the queue length processes.

Similarly,

$$W^C_{i,j}(t) - W^M_{i,j}(t) \leq 1.$$

This implies that $\forall\, t \geq 0, \bar{W}^C_{i,j}(t) = \bar{W}^M_{i,j}(t)$.

Note that:

$$\sum_{j=1}^{N-1} \bar{T}^C_{i,j}(t) = \sum_{j=1}^{N-1} \bar{T}^M_{i,j}(t).$$

This implies that:

$$\bar{I}^M_1(t) = \bar{I}^C_1(t).$$

Hence at station 1, processes in the fluid limit under the SFCR-Π policy are identical to those of the MQN-Π policy. In addition since the fluid level processes $\bar{Q}^M_{i,j}(t)$ are identical to $\bar{Q}^C_{i,j}(t)$ and the allocation processes for each class, $\bar{T}^M_{i,j}(t)$ are identical to $\bar{T}^C_{i,j}(t)$, under the SFCR-Π policy the system in the fluid limit also satisfies the non-idling criterion (2.2.9).

Note that we have established that the fluid limit processes under the SFCR-Π policy converge *pointwise* to the fluid limit processes of the MQN-Π policy. We still need to show that the fluid limits of the queue length under the SFCR-Π policy converge u.o.c to the fluid limits of the queue length under the MQN-Π policy.

We know that for the MQN-Π policy:

$$\lim_{n\to\infty} \frac{Q^M_{i,j}(nt)}{n} = \bar{Q}(t) \text{ u.o.c a.s.}$$

We have also shown that $\forall\, t \geq 0$,

$$0 \leq Q^C_{i,j}(t) - Q^M_{i,j}(t) \leq 1.$$

We need to show the following lemma to claim that the fluid model of the UMQN under the SFCR-Π policy is identical to the fluid model of the UMQN under the MQN-Π policy. Let $\bar{Q}(t)$ be the fluid limit of the queue length process for the UMQN under the MQN-Π policy.

Lemma 3.2.4.

$$\lim_{n\to\infty} \frac{Q^C_{i,j}(nt)}{n} = \bar{Q}(t) \ \textit{u.o.c a.s.}$$

Proof. Let $Q^{C,n}(t) = \dfrac{Q^C_{i,j}(nt)}{n}$ and $Q^{M,n}(t) = \dfrac{Q^M_{i,j}(nt)}{n}$.

Consider any compact set T. We need to show that $Q^{C,n}(t) \to \bar{Q}(t)$ u.o.c a.s. That is we need to show that:

$$\lim_{n\to\infty} \sup_{t\in T} \|Q^{C,n}(t) - \bar{Q}(t)\| = 0.$$

Note that:

$$\begin{aligned}
\lim_{n\to\infty} \sup_{t\in T} \|Q^{C,n}(t) - \bar{Q}(t)\| &= \lim_{n\to\infty} \sup_{t\in T} \|Q^{C,n}(t) - Q^{M,n}(t) + Q^{M,n}(t) - \bar{Q}(t)\| \\
&\leq \lim_{n\to\infty} \sup_{t\in T} \{\|Q^{C,n}(t) - Q^{M,n}(t)\| + \|Q^{M,n}(t) - \bar{Q}(t)\|\} \\
&\leq \lim_{n\to\infty} \sup_{t\in T} \{\frac{1}{n} + \|Q^{M,n}(t) - \bar{Q}(t)\|\} \\
&\leq \lim_{n\to\infty} \frac{1}{n} + \lim_{n\to\infty} \sup_{t\in T} \{\|Q^{M,n}(t) - \bar{Q}(t)\|\} \\
&\leq 0.
\end{aligned}$$

Therefore

$$\lim_{n\to\infty} Q^{C,n}(t) = \bar{Q}(t) \text{ u.o.c a.s.}$$

That is,

$$\lim_{n\to\infty} \frac{Q^C_{i,j}(nt)}{n} = \bar{Q}(t) \text{ u.o.c a.s.}$$

□

The u.o.c convergence of the scaled versions of the processes $D^C_{i,j}(t)$ and $T^C_{i,j}(t)$ can be established by an analogous argument.

Lemma (3.2.3) allows us to follow the same reasoning for all other stations, thus permitting us to claim that the fluid models for the system under the SFCR-Π policy and the MQN-Π policy are identical.

Note that:

$$T^C_{i,j}(t) \leq D^C_{i,j}(t) \leq T^C_{i,j}(t) + 1.$$

Hence, in the fluid limit:

$$\lim_{n\to\infty} \frac{T^C_{i,j}(nt)}{n} \leq \lim_{n\to\infty} \frac{D^C_{i,j}(nt)}{n} \leq \lim_{n\to\infty} \frac{T^C_{i,j}(nt)+1}{n}.$$

Therefore

$$\bar{D}^C_{i,j}(t) = \bar{T}^C_{i,j}(t).$$

3.2.4 The Fluid Model

The fluid model equations for the SFCR are listed below: $\forall\ i \in \{2 \ldots N-1\}$ and $j \in \{1, \ldots, N-i\}$,

$$\begin{aligned}
\bar{A}_{1,j}(t) &= \alpha_{1,j} t && (3.2.1)\\
\bar{A}_{i,j}(t) &= \bar{D}_{i-1,j+1}(t) && (3.2.2)\\
\bar{Q}_{i,j}(t) &= \bar{Q}_{i,j}(0) + \bar{A}_{i,j}(t) && (3.2.3)\\
&\quad -\bar{D}_{i,j}(t) + \bar{D}_{i-1,j+1}(t) &&\\
\bar{D}_{i,j}(t) &= \bar{T}_{i,j}(t) && (3.2.4)\\
\bar{I}_i(t) &= t - \sum_{j=1}^{N} \bar{T}_{i,j}(t). && (3.2.5)
\end{aligned}$$

Note that we have dropped the superscript C in referring to the various quantities involved. This is because for the remainder of this section we shall only be dealing with one fluid model. In addition, since the processes in the SFCR and the MQN-Π policy, converge to the same fluid limit, the SFCR model satisfies the following fluid model equation for non-idling:

At each station i, $\bar{I}_i(t_2) - \bar{I}_i(t_1) > 0$ for all $t_2 > t_1 \geq 0$, if and only if:

$$\bar{W}_i(t) = 0,\ \forall\, t\ \in [t_1, t_2). \tag{3.2.6}$$

Note that additional constraints will be needed to enforce the scheduling policy Π. We are now prepared to prove theorem 3.2.1.

Proof. The fluid model is a special case of the one analyzed by Dai and Weiss (section 6, [10]). Dai and Weiss show that this fluid model is stable under the

UTCs. As mentioned in section (2.3.1), it can be shown that this network is weakly stable when the UTCs are less than or equal to one. This implies the rate stability of the queueing network as shown by Chen in [6]. This completes the proof of theorem 3.2.1. □

Remark 3.2.3. Dai and Weiss [10] show that the fluid model is stable. This is a stronger notion of stability than weak stability (for further details see [9]). According to Dai's result on fluid models [8], the SFCR queueing network would be positive Harris recurrent, if it operated under a stationary policy. However, the MQN-Π policy we have used to eliminate idling is not stationary and hence the stronger result is not directly applicable.

3.3 Work in Process (WIP) Optimal Policy for SFCR

In this section our objective is to find the policy which minimizes long run average WIP for the SFCR. The expression for the long run average WIP ($\bar{L}$) along a sample path ω is:

$$\bar{L}(\omega) = \lim_{T\to\infty} \frac{\int_0^T |Q(t,\omega)| \cdot dt}{T}.$$

Theorem 3.3.1. *For any sample path ω all non-idling policies have the same long run average WIP $\bar{L}(\omega)$, in the SFCR queueing network when all the downstream stations are empty initially.*

Proof. In the proof we suppress the ω notation for simplicity. It is to be noted

that at every integer unit of time, an empty vehicle arrives at station 1. We also assume that all downstream stations are empty at time 0. We define the WIP contribution of a job as the time spent in the system by this job. Note that this is equal to the area under the WIP profile contributed by this particular job. For example a job which arrives in the system at time t_a and exits the system at time t_d has a WIP contribution of $(t_d - t_a)$.

At any integer time t, let there be M jobs in the system at station 1. Consider the k^{th} and the $(k+1)^{st}$ $(k+1 < M)$ job in the order in which the jobs are to be served. Let the destinations of these two jobs be stations l and m respectively. We first consider the case that $l < m$.

The remaining amount of time job number k spends in the system is $k + l$. Similarly the amount of time job number $k+1$ spends in the system is $k+1+m$. Let the WIP contribution of all jobs that have already finished service at station 1 at time t be L_p. Let the WIP contribution of jobs waiting at station 1 up to job number k be L_{k-1}. Similarly let the WIP contribution of all jobs after $k+1$ be L_{k+2}. Hence the total WIP at time t can be calculated as:

$$L_p + L_{k-1} + (t - t_{a,k}) + (t - t_{a,k+1}) + (k + l) + (k + 1 + m) + L_{k+2} \text{ units,}$$

where $t_{a,k}$ and $t_{a,k+1}$ are the arrival times of k^{th} and $(k+1)^{st}$ jobs.

Now let us switch the order of service for just these two jobs. That is the destination of the new k^{th} job would be m and the destination of the new $(k+1)^{st}$ job would be l. The new total WIP can be calculated as:

$$L_p + L_{k-1} + (t - t_{a,k}) + (t - t_{a,k+1}) + (k + m) + (k + 1 + l) + L_{k+2} \text{ units.}$$

This is exactly the same as the total WIP prior to switching. Hence this switching of jobs did not impact the long run average WIP. But note that this is a completely new policy. This switch also does not impact the WIP contributions of jobs prior to these two jobs or jobs after these two. Using standard interchange arguments, it can be shown that any non-idling policy can be obtained from any other non-idling policy by a series of finite pairwise interchanges like these. Hence the WIP contribution of jobs does not change regardless of the order of service and hence all non-idling policies are identical with respect to long run average WIP.

Also note that by symmetry the case that $l > m$ is identical. If $l = m$, it does not matter which job we choose. □

3.4 Optimal Draining Policy for Fluid Network

The fluid model for the SFCR under a scheduling policy Π, as we have shown earlier, is the same as that of the unidirectional ring type MQN under the MQN-Π policy. In this section we find the policy which minimizes the instantaneous amount of fluid in the SFCR network at every point of time. The solution to this problem also minimizes the long run average fluid in the network [7].

Theorem 3.4.1. *The Shortest Requested Travel Time (SRTT) first policy minimizes the draining time and the average WIP for the fluid model of the N station SFCR.*

Proof. Let the total amount of fluid in the network at time t be $\bar{W}(t)$. Also let $W_j(t)$ be the total amount of fluid at station j at time t. At station 1,

$$\bar{W}_1(t) = \sum_{j=1}^{N-1} \bar{Q}_{1,j}(0) + \sum_{i=1}^{N-1} \alpha_{1,i} \cdot t - \sum_{j=1}^{N-1} \bar{D}_{1,j}(t).$$

At any other station k,

$$\bar{W}_k(t) = \sum_{j=1}^{N-k} \bar{Q}_{k,j}(0) + \sum_{j=1}^{N-k} \bar{D}_{k-1,j+1}(t) - \sum_{j=1}^{N-k} \bar{D}_{k,j}(t).$$

Hence,

$$\begin{aligned} \bar{W}(t) &= \sum_{k=1}^{N-1} \bar{Q}_k(t) \\ &= \sum_{i=1}^{N-1} \sum_{j=1}^{N-i} \bar{Q}_{i,j}(0) + \sum_{i=1}^{N-1} \alpha_{i,i} \cdot t - \sum_{k=1}^{N-1} \bar{D}_{k,1}(t). \end{aligned}$$

It is easier to work in terms of derivatives rather than the actual workload values. The derivative of $\bar{W}(t)$ with respect to t:

$$\begin{aligned} \dot{\bar{W}}(t) &= \sum_{k=1}^{N-1} \dot{\bar{Q}}_k(t) \\ &= \sum_{i=1}^{N-1} \alpha_{1,i} - \sum_{k=1}^{N-1} \dot{\bar{D}}_{k,1}(t) \\ &= \sum_{i=1}^{N-1} \alpha_{1,i} - \sum_{k=1}^{N-1} \dot{\bar{T}}_{k,1}(t). \end{aligned}$$

To minimize the value of $\bar{W}(t)$, the value of $\dot{\bar{T}}_{k,1}(t)$ for all values of k should be set to 1 as long as $\bar{Q}_{k,1}(t)$ is not empty. The total processing rate of fluid

would be maximized by working on fluid of the non-empty buffer with least stations left to visit. For further details refer to section 4.3.

Hence the optimal policy is:

- At each station j, process the buffer belonging to the class which immediately exits the station (class $(j, 1)$) as long as it is non-empty.
- If this buffer empties allocate a fraction of the station's processing rate to keep it empty. This fraction is decided by the input rate to this buffer.
- Allocate the remaining fraction of the station's processing rate to the class of jobs, which, would exit at the next station.
- If this is station $N - 1$ allocate all capacity available to keep the buffer empty.
- As classes empty, work on classes with jobs remaining that have the least number of steps remaining to exit the system.

□

In chapter 4 we provide an example of a fluid network with no external input but with an initial amount of fluid in all buffers, where there are policies which are not optimal with respect to WIP. However in the absence of initial amounts of fluid at downstream buffers, all policies optimize WIP for the SFCR.

Chapter 4

General Feed-forward Cambridge Ring

In this chapter, we extend the previous model discussed by allowing external arrivals to all stations. We label this network the *General Feed-forward Cambridge Ring (GFCR)*. We establish the throughput optimality of all non-idling policies in the GFCR using a similar approach to the one used in the previous chapter. The optimal policy with respect to WIP in the fluid model of the GFCR is then identified. We proceed to show using examples that the WIP optimal policy as suggested by the fluid model may not be optimal for the GFCR with respect to long run average WIP. The general holding cost minimization problem for this fluid network is also formulated and the optimal solution of the holding cost problem for a simple example is presented.

4.1 Detailed Model Description

In this section we describe the GFCR with N stations and show how it fits into the UMQN framework discussed earlier. In the GFCR, all stations except station N may receive external arrivals. We assume that none of these arrivals request a destination past station N. Thus jobs arriving to station i can request any destination from $i+1, \ldots, N$. As in the previous case this implies

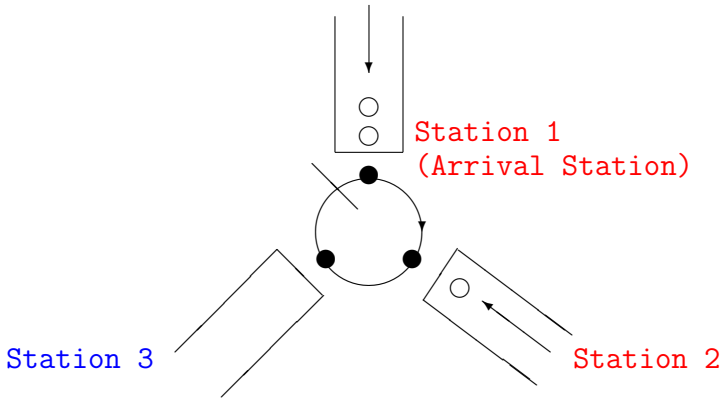

Figure 4.1: General Feed-forward Cambridge Ring

that an empty vehicle arrives to station 1 at every integer point of time. An example of this network with three stations is shown in figure 4.1.

As in the SFCR, this problem can be described in the multiclass framework developed in the previous section. We consider the UMQN described in chapter 3 and in this case we let all stations except station N receive external arrivals. Station N cannot receive external arrivals and it serves only as a destination station. At each station every job requests a service of one unit of time. As in the previous chapter, a job of type (or class) (j, k) is currently at station j with k steps remaining to reach its destination station. Thus at station i jobs can be grouped into one of potentially $N - i$ job classes. This leads to a total of $\frac{(N-1)\cdot(N-2)}{2}$ potential job classes.

The CR restrictions force service to begin at integer points of time and station 1 is free to begin service on an external arrival at every integer point of time. These restrictions enforce idling as explained in section 2.1. Again we label the policy we use to resolve contention for service Π. The UMQN described above, operating under the CR restrictions under the HL scheduling policy Π is said to be operating under the GFCR-Π policy.

4.1.1 The Queueing Network Equations

The queueing network equations for the UMQN operating under the GFCR-Π policy are almost identical to that of the SFCR except for the fact that external arrivals occur to all stations but station N. We use the same terminology as in the previous chapter, including the superscripts C and M. The queueing network equations under the GFCR-Π policy are the same as (2.1.1)-(2.1.8). However we impose the restriction that:

$$D^C_{0,j}(t) \;=\; 0. \tag{4.1.1}$$

This condition is needed to enforce the fact that station N is simply a destination station. We have relaxed the condition from the SFCR that $E_{i,j}(t) = 0, \forall\, i > 1$, and permit external arrivals to all classes. Under the CR restrictions the non-idling condition is the same as (2.1.9a) and (2.1.9a). We also need to enforce the the scheduling policy Π and the operational constraint that service at any station can only begin at integer points of time.

4.2 Stability Analysis

In this section we show that all non-idling policies Π are throughput optimal for the UMQN operating under the GFCR restrictions. Consider the GFCR with N stations described above. The corresponding UMQN is shown in figure 4.2. As in the previous case we state our main result here.

Theorem 4.2.1. *In the GFCR all non-idling scheduling policies are throughput optimal.*

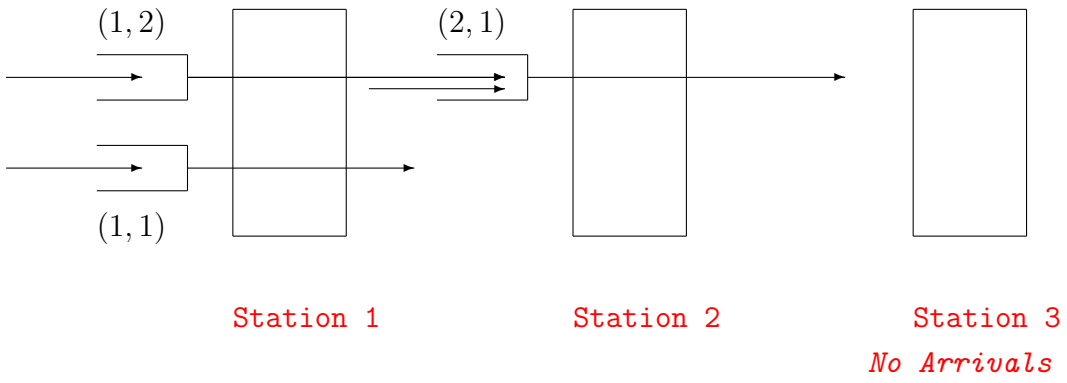

Figure 4.2: Multiclass Version of GFFR

Let $\alpha_{i,j}$ (≥ 0) be the arrival rate to class (i, j). The UTCs for the UMQN are:

For station 1,

$$\sum_{j=1}^{N-1} \alpha_{1,j} < 1.$$

For every station, $i \in \{2, \ldots, N-1\}$,

$$\sum_{j=1}^{N-i} (\alpha_{i,j} + \alpha_{i-1,j+1}) < 1.$$

The scheduling policy we compare the GFCR-Π policy to is identical to the one described in section 3.2.2. We do not describe the policy again in this section. For the UMQN operating under the GFCR-Π policy, due to the presence of external arrivals, an arrival can occur at a non-integer point of time to any station except N. As earlier we refer to this policy as the MQN-Π policy. This policy ensures that all stations have the same choice of jobs to work on at all integer points of time as the GFCR-Π policy.

Lemma 4.2.2. *For all job classes (i,j), such that $i \in \{1,\ldots,N-1\}$, $j \in \{1,\ldots,N-i\}$ and $\forall\ t \geq 0$:*

$$0 \leq T^M_{i,j}(t) - T^C_{i,j}(t) \leq 1.$$

Remark 4.2.1. Note that in the UMQN operating under the MQN-Π policy, pre-emption can occur at any station. This does not occur in the SFCR as downstream stations receive only internal arrivals and only one internal arrival can occur in the interval between two consecutive integer points of time.

Proof. This claim has already been proven in chapter 3 for all classes belonging to station 1 of the SFCR. Since station 1 of the GFCR receives only external arrivals, it is identical to station 1 of the SFCR. Hence by lemma 3.2.2,

$$0 \leq T^M_{1,j}(t) - T^C_{1,j}(t) \leq 1,\ \forall\, j \in\ \{1,\ldots,N-1\},\ \forall\, t \geq 0.$$

Consider the MQN-Π policy as applied to station i. That is, at station i service starts on the first job that arrives to an empty station i and is pre-empted if necessary at the next integer unit of time. Whenever at a non-integer time t under the MQN-Π policy, a job has to be chosen to start service, it is chosen according to the scheduling policy Π applied to the jobs present at that time. At time $\lceil t \rceil$ service on this job may be pre-empted if another job with higher priority under the GFCR-Π policy is waiting for service at station i. We prove this lemma for station 2 and show that it is true for any station in the UMQN.

Let ω be a fixed but arbitrary sample path of external arrivals. Any arrival that occurs to an empty station 2, can either be an external arrival or an

internal arrival. Consider the first non-integer time, when an arrival (internal or external) occurs to an empty station 2. This time need not be identical under the MQN-Π and GFCR-Π policies. If it is an internal arrival which occurs to this empty station under the MQN-Π policy, then under the GFCR-Π policy this arrival may not occur until the next integer unit of time. However the event of an arrival facing an empty station 2 at a non-integer time will occur under the MQN-Π policy earlier than or at the same time as under the GFCR-Π policy. Let τ_{B2} be the first non-integer time under the MQN-Π policy that an arrival to some class say $(2, k)$, faces an empty station 2. Let us call this arrival m_k.

If this arrival m_k is an internal arrival from class $(1, k+1)$, the way the MQN-Π policy is setup at station 1 ensures that this arrival will occur under the GFCR-Π policy at time $\lceil \tau_{B2} \rceil$. Under the GFCR-Π policy, service can commence on this job at station 2 (if it is the highest priority job in queue) at time $\lceil \tau_{B2} \rceil$. If on the other hand, this arrival m_k is an external arrival, then it occurs under both policies at the same time. Hence under both the GFCR-Π policy and the MQN-Π policy the UMQN can choose to process job m_k at time $\lceil \tau_{B2} \rceil$. Let δ_{B2} be the earliest time after τ_{B2}, under the GFCR-Π policy when the station 2 is idle again. Hence the order in which the jobs receive service under both policies is the same at all stations.

We now compare the times spent working on each class under the GFCR-Π policy and the MQN-Π policy. As at station 1 we analyze the difference $T^M_{2,k}(t) - T^C_{2,k}(t)$, $\forall\, k \in \{2, \ldots, N-2\}$ over $(\tau_{B2}, \delta_{B2}]$. The two cases we need

to consider are:

1. Arrival m_k is an internal arrival which has just completed service from station 1.

2. Arrival m_k is an external arrival to class $(2, k)$.

Case 1:

If under the MQN-Π policy at station 2, service begins on an internal arrival, m_k at time τ_{B2}, (before this internal arrival occurs under the GFCR-Π policy) the time spent serving class $(2, k)$ starts increasing. The difference $T^M_{2,k}(\tau_{B2}) - T^C_{2,k}(\tau_{B2})$ is $\lceil \tau_{B2} \rceil - \tau_{B2}$. Then at time $\lceil \tau_{B2} \rceil$, internal arrival m_k occurs under the GFCR-Π policy. Once this internal arrival occurs under the GFCR-Π policy, $T^M_{2,k}(t) - T^C_{2,k}(t)$ remains constant until this job starts service under the GFCR-Π policy.

Once the internal arrival occurs, under the MQN-Π policy station 2 operates similar to station 1. At time $\lceil \tau_{B2} \rceil$, under the GFCR-Π policy the UMQN may have a choice of jobs on which to commence service. If the internal arrival m_k (currently in service under the MQN-Π policy) gets the highest priority under the GFCR-Π policy, service continues under the MQN-Π policy. Under the GFCR-Π policy, the system commences service to this job at time τ_{B2}. Otherwise the job is preempted under the MQN-Π policy by the job of the class with highest priority under the GFCR-Π policy, say $(2, l)$. The difference $T^M_{2,k}(t) - T^C_{2,k}(t)$, remains constant until the partially completed job m_k resumes

service. This occurs the next time class $(2, k)$ becomes the highest priority class under the GFCR-Π policy. This will occur at an integer point of time before the end of the current busy period in the MQN-Π policy. Note that at time $\lceil \tau_{B2} \rceil$, there is only a single class (class $(2, k)$) with $T^M_{2,k}(t) - T^C_{2,k}(t) > 0$ and this difference is less than or equal to one.

The rest of the proof in this case runs exactly as in station 1. That is, until the end of a busy period under the MQN-Π policy, the difference gets transferred between the classes but never increases beyond one. The busy period under the MQN-Π policy will end before the busy period under the GFCR-Π policy. When under the GFCR-Π policy the busy period ends, $(T^M_{2,k}(t) - T^C_{2,k}(t), \forall\, k \in \{1, \ldots, N-2\})$ decrease until they becomes zero at the next integer unit of time, unless another arrival occurs before then. In this case the end of the busy period argument made for station 1 in the proof of lemma 3.2.2 applies. Then there will be two classes at station 2 with positive differences in time served but the total differences will still be less than one.

Case 2:

If this job m_k, is an external arrival, then it occurs at the same time regardless of the policy. Under the MQN-Π policy, service for this job commences immediately. As in case 1, $T^M_{2,k}(t) - T^C_{2,k}(t)$, starts increasing until at time $\lceil \tau_{B2} \rceil$, the difference becomes, $\lceil \tau_{B2} \rceil - \tau_{B2}$.

At time $\lceil \tau_{B2} \rceil$, if there is a higher priority job class in the queue under the GFCR-Π policy, the job m_k gets preempted under the MQN-Π policy. Under

the GFCR-Π policy, the job m_k, then resumes service when it becomes the highest priority job.

Hence the sequence of events that occur after the next integer unit of time is exactly the same regardless of whether the arrival is internal or external. The rest of the proof is exactly identical to the proof in the previous chapter.

It is to be noted that this proof is similar at all stations because at each station:

- The order in which jobs complete processing is identical under both the GFCR-Π and the MQN-Π policies.
- Any internal arrival under both policies occurs to the station within the same integer unit of time. This enables both policies to be able to process the same job at every integer point of time.

□

The important feature of the MQN-Π policy is that at any downstream station, external arrivals can be treated exactly the same way as internal arrivals. This is because the time of arrival of the internal arrival t does not matter as long as under the GFCR-Π policy, it arrives before $\lceil t \rceil$.

4.2.1 Illustrative Example

An illustrative example for the GFCR with three stations (figure 4.2) is provided to compare the operation of the network under a specific GFCR-Π policy and the corresponding MQN-Π policy. The external arrival process consists of

only four arrivals as shown in table 4.1. We call these four arrivals, jobs 1, 2, 3 and 4. Job 1 belongs to class $(1,1)$ and arrives at time 0.3. Job 2 belongs to class $(2,1)$ and arrives at time 0.7. Job 3 belongs to class $(1,2)$ and arrives at time 0.9. Job 4 belongs to class $(1,1)$ and arrives at time 1.3. Since we only use this example to compare the policies over a small time window, the arrival times for these jobs were chosen to highlight events that were discussed in the previous section. For example, the first arrival faces an empty station and hence under the GFCR-Π policy, service cannot begin until time 1 whereas under the MQN-Π policy, service would commence immediately.

In this example, the GFCR-Π policy is a static buffer priority (SBP) policy at station 1. At station 1, class $(1,2)$ has priority over class $(1,1)$. Since there is just one class at station 2, we simply follow the FIFO policy. Service under the GFCR-Π policy at each station begins on the job belonging to the class with highest priority at every integer unit of time. The corresponding MQN-Π policy is the same static buffer priority policy described above. However for an arrival that faces an empty station, service begins on this job immediately rather than at the next integer unit of time. At the next integer unit of time if there are any jobs with higher priority, service on the current job is preempted in favor of the higher priority job. Under the MQN-Π policy, the system resumes service on this job when service commences on this job under the GFCR-Π policy.

Note that any preemption referred to occurs only under the MQN-Π policy. There is no preemption under the GFCR-Π policy. At time 0.3 an external

arrival (job 1) occurs to class $(1,1)$. Under the MQN-Π policy, service commences on this job immediately. At time 0.7, an external arrival (job 2) occurs to class $(2,1)$. Again under the MQN-Π policy at station 2, service commences on this job immediately.

At time 0.9, an external arrival (job 3) occurs to class $(1,2)$. During the interval $(0.3,1)$, under the GFCR-Π policy the network remains idle at both stations while under the MQN-Π policy service on jobs 1 and 2 is in progress. At the next integer unit of time (1.0), under the GFCR-Π policy service starts on the highest priority jobs available at stations 1 and 2. Hence service begins on job 3 at station 1 and job 2 at station 2. Under the MQN-Π policy at time 1.0, job 1 with 0.3 units of service time remaining is preempted by a job of class $(1,2)$ (job 3). However at station 2, there is no other higher priority job waiting and hence service continues on job 2. The differences in processing times for the classes at stations 1 and 2 are as shown in table 4.1.

At time 1.4, there is an external arrival (job 4) of class $(1,1)$. In our example this is the last external arrival. At time 1.7, under the MQN-Π policy service is completed on job 2. Under the GFCR-Π policy, 0.3 units of service time remain on this job at this time.

At time 2, under the GFCR-Π policy service is completed on job 3 and job 2. The next job to commence service under the GFCR-Π policy is of class $(1,1)$ (job 1). Under the MQN-Π policy service on job 3 is also completed at station 2. Also service is resumed on job 1 at station 1, while under the GFCR-Π policy service commences on this job. At this point of time under these two

policies the network has spent the same amount of time working on classes $(1,2)$ and $(2,1)$, while on class $(1,1)$, the MQN-Π policy is ahead by 0.7 time units. This difference remains the same as at time 1.0.

The job completed at time 2 from class $(1,2)$ (job 3) becomes an internal arrival to class $(2,1)$ at the same time under both policies. Under both policies service starts on this job immediately at time 2.0. Hence the amount of time spent processing jobs at station 2 is equal under both policies until time 3.0. At time 2.3, job 1 completes service under the MQN-Π policy and departs from the system. This job gets completed under the GFCR-Π policy at time 3.0. Under the MQN-Π policy service begins on job 4 at time 2.3 and completes service at time 3.3. Under the GFCR-Π policy service starts on job 4 at time 3.0 and completes it at time 4.0. At this time both systems have spent the same amount of time working on jobs in all classes.

The purpose of this example is to compare the functioning of the UMQN under both policies and to demonstrate that the time spent working on the various classes under each policy differs by less than one at any point of time. Table 4.1 shows the various events which occur and a comparison of the time spent working on the various classes at different points of time.

4.2.2 The Fluid Model for the Cambridge Ring

Consider again a fixed but arbitrary sample path ω. Along this sample path $T^C_{i,j}(t) - T^M_{i,j}(t) \leq 1$ (Lemma 4.2.2). As in the previous case, $D^C_{i,j}(t) - D^M_{i,j}(t) = \lfloor T^C_{i,j}(t) \rfloor - \lfloor T^M_{i,j}(t) \rfloor$, which implies that $D^C_{i,j}(t) - D^M_{i,j}(t) \leq 1$.

Table 4.1: Example to compare operation of GFCR and MQN-Π policies

Event	time(t)	$T^M_{1,1}(t) - T^C_{1,1}(t)$	$T^M_{1,2}(t) - T^C_{1,2}(t)$	$T^M_{2,1}(t) - T^C_{2,1}(t)$
External Arrival (job 1) to class $(1,1)$	0.3	0	0	0
External Arrival (job 2) to class $(2,1)$	0.7	0.4	0	0
External Arrival (job 3) to class $(1,2)$	0.9	0.6	0	0.2
Job 1 is preempted by job 3	1.0	0.7	0	0.3
Service to job 2 commences under GFCR-Π policy				
System snapshot	1.3	0.7	0	0.3
External Arrival (job 4) to class $(1,1)$	1.4	0.7	0	0.3
Job 2 completes service under MQN-Π policy	1.7	0.7	0	0.3
System snapshot	1.8	0.7	0	0.2
Service to job 3 is complete under both policies	2.0	0.7	0	0
Internal Arrival to class $(2,1)$,				
Service to job 1 resumes under MQN-Pi policy				
Service to job 1 starts under GFCR-Π policy				
System snapshot	2.1	0.7	0	0
Departure of first class $(1,1)$ job under MQN-Π policy	2.3	0.7	0	0
Departure of first class $(1,1)$ job under GFCR-Π policy	3.0	0.7	0	0
Departure of job 4 under MQN-Π policy	3.3	0.7	0	0
System snapshot	3.5	0.5	0	0
Departure of job 4	4.0	0	0	0
under GFCR-Π policy				

This implies that $A^C_{i,j}(t) - A^M_{i,j}(t) \leq 1$, since for all $i > 1$,

$$A^C_{i,j}(t) - A^M_{i,j}(t) = D^C_{i-1,j+1}(t) - D^M_{i-1,j+1}(t).$$

At station 1, $A^C_{i,j}(t) - A^M_{i,j}(t)$ is zero. The external arrival processes are identical under both policies and hence do not play a role in the difference. In the GFCR, the initial station acts as a synchronizing station.

Thus along any sample path ω, $A^M(t) - A^C(t)$, $T^M(t) - T^C(t)$ and $D^M(t) - D^C(t)$ $\forall\, i, j$ are less than or equal to one. Thus as argued in section 3.2.3, the differences of the corresponding fluid limits are zero.

This shows that in the fluid limit, the various processes for the UMQN under the GFCR-Π policy and MQN-Π policy are identical pointwise. Also as shown in lemma 3.2.4,this implies that the processes of the UMQN under the GFCR-Π policy converge to the fluid limit processes under the MQN-Π policy uniformly on compact sets along any sample path ω.

Since the fluid model processes for the GFCR-Π policy (which occasionally "enforces" idling) are identical to the fluid model processes of a non-idling policy, the non-idling condition for the fluid model of the GFCR-Π policy is the same as that of the MQN-Π policy.

Also as argued in section 3.2.3, $\bar{D}^C_{i,j}(t) = \bar{T}^C_{i,j}(t)$. The fluid model equations for

the GFCR are listed below: $\forall\ i \in \{1 \ldots N\},\ j \in \{1, \ldots, N-i\}$ and $t \geq 0$:

$$\begin{aligned}
\bar{A}_{i,j}(t) &= \alpha_{i,j}t + \bar{D}_{i-1,j+1}(t) && (4.2.1)\\
\bar{Q}_{i,j}(t) &= \bar{Q}_{i,j}(0) + \bar{A}_{i,j}(t) - \bar{D}_{i,j}(t) && (4.2.2)\\
\bar{W}_i(t) &= \sum_{j=1}^{N-i} \bar{Q}_{i,j}(t) && (4.2.3)\\
\bar{D}_{i,j}(t) &= \bar{T}_{i,j}(t) && (4.2.4)\\
\bar{I}_i(t) &= t - \sum_{j=1}^{N} \bar{T}_{i,j}(t) && (4.2.5)\\
\bar{D}_{0,j}(t) &= 0. && (4.2.6)
\end{aligned}$$

The non-idling condition for this fluid model is that at each station i, $\bar{I}_i(t_2) - \bar{I}_i(t_1) > 0$ for all $t_2 > t_1 \geq 0$, if and only if:

$$\bar{W}_i(t) = 0,\ \forall\, t\ \in [t_1, t_2). \tag{4.2.7}$$

As in the SFCR, additional constraints will be needed to enforce the actual GFCR-Π policy. The proof of theorem 4.2.1 then follows from an argument exactly analogous to the proof of theorem 3.2.1.

4.3 WIP Optimal Draining Policy for Fluid Network

Let $\bar{W}(t)$ be the total amount of fluid in the network at time t and $\bar{Q}_k(t)$ be the total amount of fluid at station k at time t. We find the global optimum by minimizing the value of $\bar{W}(t)$ for every $t \geq 0$.

At station 1,

$$\begin{aligned}\bar{Q}_1(t) &= \sum_{j=1}^{N-1} \bar{Q}_{1,j}(0) + \sum_{j=1}^{N-1} \bar{A}_{1,j}(t) - \sum_{j=1}^{N-1} \bar{D}_{1,j}(t) \\ &= \sum_{j=1}^{N-1} \bar{Q}_{1,j}(0) + \sum_{j=1}^{N-1} \alpha_{1,j}(t) - \sum_{j=1}^{N-1} \bar{D}_{1,j}(t).\end{aligned}$$

At any other station i $(1 < i < N)$,

$$\begin{aligned}\bar{Q}_i(t) &= \sum_{j=1}^{N-i} \bar{Q}_{i,j}(0) + \sum_{j=1}^{N-i} \bar{A}_{i,j}(t) - \sum_{j=1}^{N-1} \bar{D}_{i,j}(t) \\ &= \sum_{j=1}^{N-i} \bar{Q}_{i,j}(0) + \sum_{j=1}^{N-i} \alpha_{i,j}(t) + \sum_{j=1}^{N-i} \bar{D}_{i-1,j+1}(t) - \sum_{j=1}^{N-1} \bar{D}_{i,j}(t).\end{aligned}$$

Hence at time t, the total amount of fluid $\bar{W}(t)$ is given by:

$$\begin{aligned}\bar{W}(t) &= \sum_{i=1}^{N} \bar{Q}_i(t) \\ &= \sum_{i=1}^{N} \sum_{j=1}^{N-i} \bar{Q}_{i,j}(0) + \sum_{i=1}^{N-1} \sum_{j=1}^{N-i} \alpha_{i,j} \cdot t + \sum_{i=1}^{N-1} \sum_{j=1}^{N-i} \bar{D}_{i-1,j+1}(t) - \sum_{j=1}^{N-1} \bar{D}_{i,j}(t) \\ &= \sum_{i=1}^{N} \sum_{j=1}^{N-i} \bar{Q}_{i,j}(0) + \sum_{i=1}^{N-1} \sum_{j=1}^{N-i} \alpha_{i,j} \cdot t - \sum_{k=1}^{N-1} \bar{D}_{k,1}(t).\end{aligned}$$

Theorem 4.3.1. *An optimal policy with respect to both work in process for the fluid model of a GFCR is a static buffer priority policy, which can be described as follows: at any station i, the order of priority is $(i,1) > (i,2) > \cdots > (i, N-i)$.*

Proof. The time derivative of $\bar{W}(t)$ is given by:

$$\begin{aligned}\dot{\bar{W}}(t) &= \sum_{i=1}^{N-1}\sum_{j=1}^{N-i}\alpha_{i,j} - \sum_{k=1}^{N-1}\dot{\bar{D}}_{k,1}(t) \\ &= \sum_{i=1}^{N-1}\sum_{j=1}^{N-i}\alpha_{i,j} - \sum_{k=1}^{N-1}\dot{\bar{T}}_{k,1}(t).\end{aligned}$$

$\dot{\bar{W}}(t)$ represents the rate of change of WIP in the fluid model. The global optimal policy is the one that minimizes $\dot{\bar{W}}(t)$ subject to the constraints specified by the fluid model equations (4.2.1)-(4.2.7).

This problem reduces to maximizing $\sum_{k=1}^{N-1}\dot{\bar{T}}_{k,1}(t)$ subject to the constraints specified by the fluid model. At any station i, $\dot{\bar{T}}_{i,1}(t)$ can attain a maximum value of 1 as long as the buffer $(i,1)$ is non-empty. It is immediately clear that if buffer $(i,1)$ is non-empty the station should spend all its time working on that buffer.

Let us define a *route* in the fluid network of the GFCR as the set of buffers that fluid arriving to class $(1,j)$ at station 1 has to pass through before it exits the system. For example $(1,1)$ is a route. Similarly $(1,2),(2,1)$ is a route. Note that every buffer belongs to exactly one route. Next we define what we refer to as the *exit buffers*. Along a route the last non-empty buffer is defined to be the exit buffer. Fluid belonging to an exit buffer along a route may exit the system at some downstream station. The set of exit buffers varies with time. If buffers of type $(k,1)$ are non-empty at time t, then they are exit buffers.

At any point of time it is the set of non-empty exit buffers that control the rate at which fluid can be drained from the system. Once an exit buffer becomes

empty it can be removed from the system and its corresponding route. The system then simply allocates the capacity necessary to keep it empty. Further at any station j, it is optimal to work on an exit buffer (j,k) $(k > 1)$ if fluid at all buffers at station j, $(j,1)\ldots,(j,k-1)$ are empty. This achieves the maximum rate at which fluid exits the network at any point of time and hence such an allocation is optimal with respect to WIP.

Once all the buffers in the fluid model empty, they remain empty as we assume that the UTCs are satisfied. If the UTCs are strictly violated, all policies have an infinite objective value. □

The above description prescribes exactly what the GFCR needs to be working on at any state and the resulting policy is the static buffer priority policy described in theorem 4.3.1. It is interesting to note that this policy is the fluid version of the SRTT policy.

Of course, theorem 4.3.1 is only interesting if there exists policies which perform worse than the SRTT policy. We use the network with three stations and fluid levels a, b and c as shown in figure 4.3 to demonstrate that not all policies are globally optimal.

We consider the fluid levels in this network under two policies - the SRTT policy and the LRTT policy. The draining time of the fluid in this network is identical under both policies but the average WIP is not. The graphs of the fluid levels under the two policies plotted against time are shown in figure 4.4. It is clear that the average WIP of the LRTT policy is higher than that of the

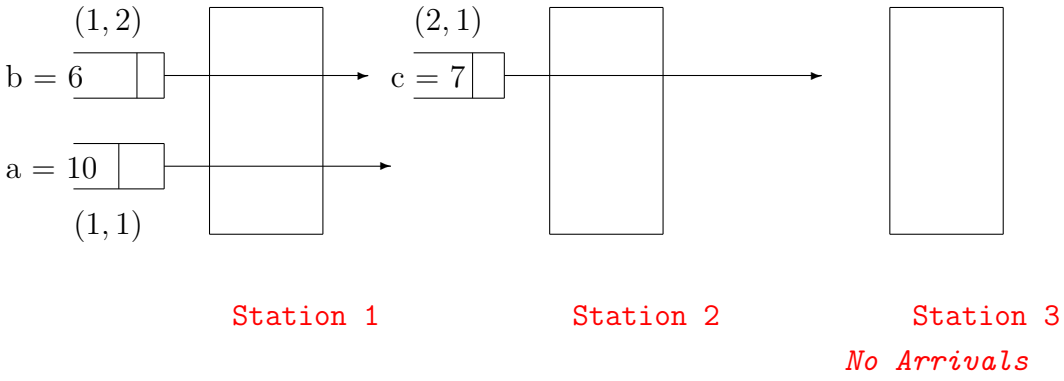

Figure 4.3: Multiclass Version of GFFR

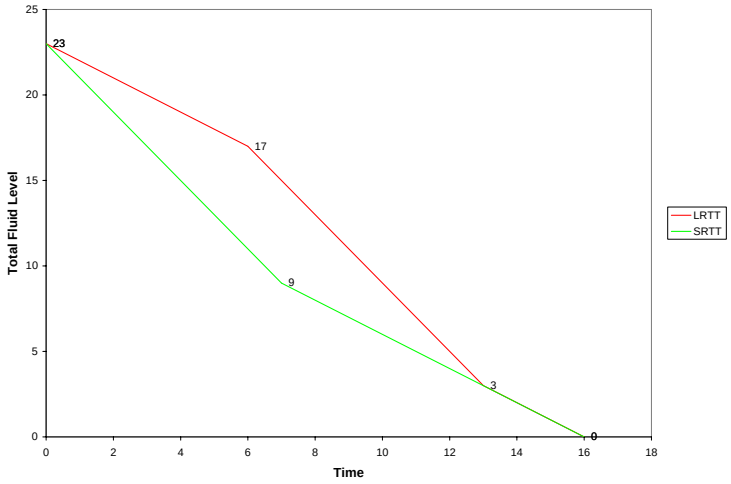

Figure 4.4: Fluid WIP Profiles under SRTT and LRTT policies

SRTT policy as the fluid levels in the LRTT policy are greater than or equal to fluid levels in SRTT policy at all times when there is a positive amount of fluid in the network.

4.4 SCLP Formulation of Holding Cost Problem for the Fluid Model

In the previous section we described the fluid optimal solution for the WIP problem. Note that minimizing WIP corresponds to minimizing fluid holding costs when the holding cost is equal for every class of fluid. It is also interesting to examine the problem with more general holding costs. In this problem we assign different holding costs for the jobs at different stations. For the GFCR network, the holding cost problem is extremely difficult to solve.

One of the approaches, suggested by Weiss [23] is to use the fluid network to set up a constrained optimization problem over any specified time horizon T. This optimization problem falls under the category of infinite dimensional mathematical programs known as separated continuous linear programs (SCLPs).

A large value of T can be used to optimize this problem to obtain a long term operating policy. If T is picked large enough for the fluid model to empty, then the resulting policy is optimal over the entire time horizon. This is because once the fluid model empties it stays empty. The fluid solution can then be used to control the operation of the GFCR. Instead of applying the fluid policy directly one may then translate the fluid optimal control policy [19] to a queueing network policy and use it to control the system.

In this dissertation we simply model our problem as an SCLP. We present the solution for a simple example of the GFCR but we do not attempt to solve the holding cost problem for a general GFCR. The objective in this problem is to minimize the holding costs incurred over some finite time horizon.

Let $c_{i,j}$ be the holding cost for class (i,j). We set $\bar{Q}_{i,j}(t)$ to be zero for all t, if class (i,j) does not exist. The fluid control problem is to choose at any point of time s in the interval $[0,T]$, optimal rates of flow, $u_{i,j}(s)(=\dot{\bar{D}}_{i,j}(s))$, which optimize the SCLP below:

$$\min \int_0^T \sum_{i=1}^{N} \sum_{j=1}^{N} c_{i,j} \bar{Q}_{i,j}(t) dt$$

subject to

$$\begin{aligned}
\dot{\bar{Q}}_{i,j}(t) &= \alpha_{i,j} + u_{i-1,j+1}(t) - u_{i,j}(t) && (4.4.1)\\
\bar{Q}_{i,j}(0) &= \bar{Q} && (4.4.2)\\
\sum_{j=1}^{N-i} u_{i,j}(t) &\leq 1 && (4.4.3)\\
\bar{Q}_{i,j}(t) &\geq 0 && (4.4.4)\\
u_{i,j}(t) &\geq 0 && (4.4.5)\\
u_{0,j}(t) &= 0, && (4.4.6)
\end{aligned}$$

where index $i \in \{1,\ldots,N-1\}$ and $j \in \{1,\ldots,N-i\}$.

The variables $u_{i,j}(s)$ are known as the control variables and the $Q_{i,j}(t)$ are known as the state variables. We now present the solution to an example of the GFCR with three stations (figure 4.2). The three job classes are $(1,1)$,

$(1,2)$ and $(2,1)$. The optimization problem simplifies to:

$$\min \int_0^T (c_{1,1}\bar{Q}_{1,1}(t) + c_{1,2}\bar{Q}_{1,2}(t) + c_{2,1}\bar{Q}_{2,1}(t))dt$$

subject to

$$\begin{aligned}
\dot{\bar{Q}}_{1,1}(t) &= \alpha_{1,1} - u_{1,1}(t) \\
\dot{\bar{Q}}_{1,2}(t) &= \alpha_{1,2} - u_{1,2}(t) \\
\dot{\bar{Q}}_{2,1}(t) &= \alpha_{2,1} + u_{1,2}(t) - u_{2,1}(t) \\
\bar{Q}_{i,j}(0) &= \bar{Q} \\
u_{1,1}(t) + u_{1,2}(t) &\leq 1 \\
u_{2,1}(t) &\leq 1 \\
\bar{Q}_{i,j}(t) &\geq 0 \\
u_{i,j}(t) &\geq 0.
\end{aligned}$$

As stated earlier this is a problem in optimal control and the goal is to determine the optimal control at time 0 given by $u(0) = (u_{1,1}(0), u_{1,2}(0), u_{2,1}(0))$ for all values of the initial state. Since this control is determined for an arbitrary initial state, these controls can be used to determine the optimal control at any time t given the state at time t. Avram et al. [3] solve this problem using the Pontryagin's Maximum Principle for a case with unequal service rates for each class. Our model is a special case of theirs as all service rates are identical. Here we present their solution modified for our problem. Note that it is evident that as long as there is fluid present at class $(2,1)$, $u_{2,1}(t)$ has to be maintained at 1. Note that once a buffer is emptied it is never optimal to

let it fill up again. Thus the nature of the optimal control is piecewise linear. This reduces the choice of flow rates to two vectors $(0, 1, 1)$ and $(1, 0, 1)$.

Case 1: $c_{1,2} < c_{1,1}$

In this case it is intuitively obvious that all the fluid from class $(1, 1)$ needs to be served first as long as it is non-empty. Hence the optimal control at time 0 is $(1, 0, 1)$. Once buffer $(1, 1)$ becomes empty it is never optimal to let it fill up again.

Case 2: $c_{1,2} > c_{1,1}$ and $c_{1,2} - c_{1,1} > c_{2,1}$

In this case the optimal control is $(0, 1, 1)$. Note that under this control, the instantaneous rate at which holding cost is reduced is maximized. This control policy is the so called myopically optimal policy [7]. In general this myopically optimal policy may not be the optimal policy with respect to holding costs.

The other case is that $c_{1,2} > c_{1,1}$ and $c_{1,2} - c_{1,1} < c_{2,1}$. In this case we are indifferent between the two available flow control rate vectors.

4.5 SRTT may not be Optimal for GFCR

In the previous section we have showed that the policy which minimizes both draining time and WIP in the fluid model is the static buffer priority policy which corresponds to the SRTT policy. In this section we show that SRTT may not be optimal for the queueing network, thus indicating the possible limitations of the use of a fluid model for scheduling purposes even in this simplified version of a CR network. We accomplish this by using a simple

example with three stations and deterministic arrival rates.

4.5.1 Counterexample

Consider the GFCR with three stations. There are three classes of jobs: $(1,1)$, $(1,2)$, $(2,1)$. Let the external arrival rate to all three classes be 0.5 jobs per unit time. Note that these arrival rates satisfy the UTCs. We assume that these external arrivals occur to the system at times $t = 2, 4, \ldots$. Further we also assume that the initial number of jobs in each class is one.

At time 0, the number in the system is 3. All vehicles are empty. Under the SRTT policy, the job at class $(1,1)$ is picked up first. Simultaneously the job at station 2 is picked up by another vehicle. Hence the WIP in the interval $[0,1)$ is equal to 3 jobs. At time 1, two jobs have left the system and hence there is now one job in the system. This job belongs to class $(1,2)$. At time 1, this job is picked up by an empty vehicle and conveyed to station 2 by time 2. At time 2, the next set of arrivals occurs raising the number in system to 4 jobs. At this time, a job of class $(1,1)$ begins service at station 1. At station 2, there are two jobs waiting for service essentially indistinguishable from each other. Service can begin on either job. Hence at time 3, one of these jobs exits the system from station 2 and another job exits the system from station 1. This leaves two jobs in the system, one of class $(2,1)$ and another of class $(1,2)$. Both of these jobs begin service at their respective stations. At time 4, the job of class $(2,1)$ exits the system from station 2 and the job of class $(1,2)$ moves to class $(2,1)$. At this point the next set of arrivals occur and the

system state is identical to the state at time 2. After this point of time the WIP oscillates between two jobs and four jobs every two units of time. Hence the long run average WIP for this system is 3 jobs.

Now let us consider the system operating under the LRTT policy. At time 0, the number in the system is 3. All stations are idle. Under the LRTT policy, the job of class $(1, 2)$ is served first. Simultaneously the job at station 2 also begins service. Hence the WIP in the interval $[0, 1)$ is equal to 3 jobs. At time 1, only one job has left the system and hence there are now two jobs in the system. At this time, the job of class $(1, 2)$ has been transferred to class $(2, 1)$ and there is only a job of class $(1, 1)$ at station 1. Hence both of these jobs can begin service at once and therefore at the end of two units of time, all three jobs leave the system. However at this point three more arrivals occur at the three classes and we are in exactly the same situation as at time 0. The long run average WIP in this case is 2.5 jobs. The WIP profile for both policies is shown in figure 4.5.

This example shows that in this simple deterministic case, the optimal policy as suggested by the fluid model is not optimal for the queueing network.

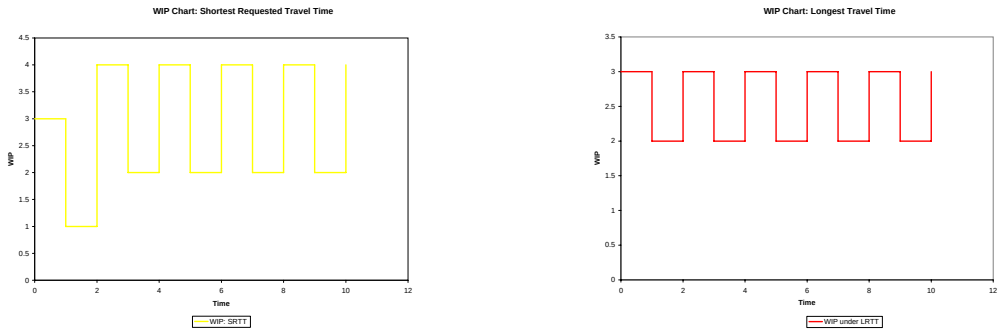

Figure 4.5: WIP Profiles under SRTT and LRTT policies

Chapter 5

Idling Policies and WIP Optimality

The purpose of this chapter is to show that an idling policy may perform better than a non-idling policy with respect to WIP for a CR. We accomplish this by considering a special case of the network described in section 5.1. We also provide an example of a GFCR in which an idling policy is optimal with respect to WIP.

Policies for the CR can be classified into two kinds:"ejective" and "non-ejective" policies. An *ejective policy* is one in which a job on a vehicle can be unloaded at an intermediate station prior to its destination to pick up a job with higher priority waiting at the intermediate station. A *non-ejective policy* is one in which a job which has been loaded on a vehicle can only be removed from the vehicle if its destination has been reached. In this chapter we only consider non-ejective policies (see section 5.1 for explanation).

5.1 Single Vehicle Case

In this section we discuss the problem of optimizing WIP in a type of CR in which there are four stations and just one vehicle. We call this problem the *Single Vehicle Cambridge Ring (SVCR)*. The vehicle moves from station to

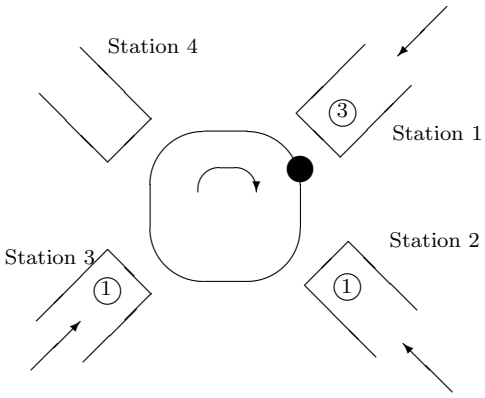

Figure 5.1: Single Vehicle CR

station. The travel time between stations is one unit of time. Since there is just one vehicle the time between visits to a particular station is four units. In our example there are just three classes of jobs in the system. Class $(1, 3)$ jobs which arrive to station 1 and request station 4 as a destination. Class $(2, 1)$ and class $(3, 1)$ are the other two job classes. It is to be observed that no job class requests a destination past station N. Hence this is a feed-forward network. This network is shown in figure 5.1.

For this special case the UTCs are:

$$\rho_1 \equiv \alpha_{1,3} \leq 1 \tag{5.1.1}$$

$$\rho_2 \equiv \alpha_{1,3} + \alpha_{2,1} \leq 1 \tag{5.1.2}$$

$$\rho_3 \equiv \alpha_{1,3} + \alpha_{3,1} \leq 1. \tag{5.1.3}$$

Due to the presence of a single vehicle jobs at any station can only start service

once every our units of time. Hence this problem does not directly fit into the multiclass view that we have used for the previous cases. It can still be modeled as a multiclass network with "open" and "closed" customers. Open customers are those who exit the system in a finite amount of time with probability 1. Closed customers do not exit the system and remain indefinitely. Bonald and Down [5] show that FIFO is throughput optimal for a "mixed" generalized Jackson network with both open and closed customers. However there is no result on the stability of a general multiclass queueing with open and closed customers. We have not attempted to find the stability conditions for this problem. Our main focus in this chapter is the WIP analysis of this model.

The WIP optimal policy for this network is the non-idling "ejective" policy listed below:

- At station 1 if there are jobs waiting, pick up a job and move it to station 2.
- At station 2 if a job is waiting, eject the job currently in service into a buffer at station 2. Pick up a job of type $(2, 1)$ and discharge it at station 3. Resume service on the ejected job whenever the vehicle arrives to station 2 and there are no jobs of class $(2, 1)$ in the buffer. At station 3 if a job is waiting, select it upon discharge of the job from station 2.
- If no jobs are waiting at station 2, transport the job of class $(1, 3)$ to its destination.

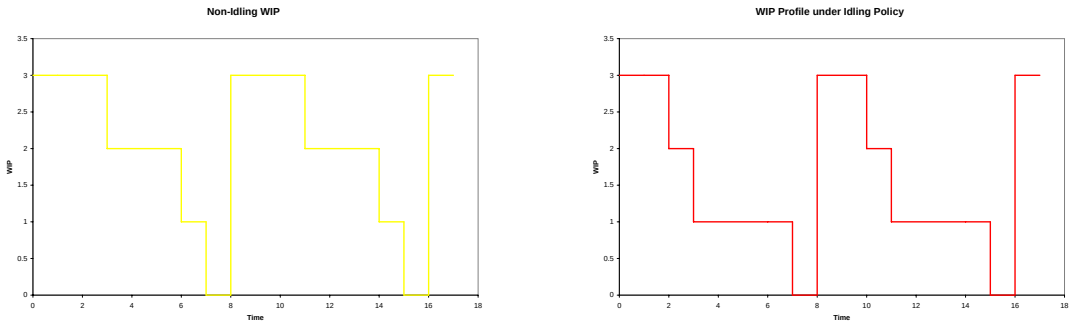

Figure 5.2: WIP Profiles under Non-idling and Idling policies

However in many applications, ejection is not permissible. Hence in this chapter we only consider "non-ejective" policies. Under "non-ejective" policies the following example demonstrates that idling may be optimal with respect to average WIP.

5.1.1 Example

Consider the SCVR with the following arrival patterns at stations 1, 2 and 3: each class receives external arrivals at times $0, 8, 16, \ldots$. Note that the implied arrival rates satisfy the UTCs. In this case it is optimal with respect to WIP to idle at station 1 at times $0, 8, \ldots$. Under a non-idling policy the average WIP in this system is 2 units, while under the idling policy the average WIP in this system is 1.5 units. The WIP chart under the two policies is shown in figure 5.2.

Consider the SVCR operating under a non-ejective policy. Since the vehicle

arrives empty to station 1 every four units, the only scheduling decision to be made is whether to load a job waiting at station 1 when the vehicle reaches station 1 or to let it travel empty to station 2. It can never be optimal with respect to WIP to idle at any of the downstream stations.

Note that there only exists a single non-ejective non-idling HL policy. We now consider a policy in which we idle at station 1 whenever there are jobs waiting at station 2. This policy is referred to as the *heuristic idling policy (HIP)*.

Lemma 5.1.1. *The long run average WIP under the HIP is less than or equal to that under the non-idling policy.*

Proof. Consider an arbitrary sample path of external arrivals. Let the vehicle be at station 1 at time $t \in \mathbb{Z}$. We also assume that there are jobs waiting at station 1. Consider the situation when there are jobs waiting at station 2.

Under the non-idling policy a job from station 1 would be loaded on to the vehicle at time t and would exit the system at time $t+3$. The number of jobs removed from the system until the next time the vehicle visits station 1 is 1. Once the job is loaded at station 1 the number in the system can decrease only at time $t+3$ and no jobs at the other stations can be removed.

Under the HIP policy, the vehicle would travel to station 2 empty. Hence at least one job will be removed from the system before the vehicle re-visits station 1, which is the same as the number of jobs removed from the system under the non-idling policy. If there are jobs present at station 3, then the number of jobs removed from the system will be 2. The job at station 2 will

be removed at time $t+2$. If there are jobs at station 3 by time $t+2$, one job will be removed by time $t+3$.

Thus whenever there are jobs present at station 2, the HIP removes at least as many or more jobs from the system than the non-idling policy. Further, under the HIP the job removed will be removed from the system with a departure time not later than a departure time under the non-idling policy. Hence the area under the WIP profile under HIP will be less than or equal to that under the WIP profile under the non-idling policy and this will happen every time there is a job at station 2 and the vehicle is in transit between stations 4 and 1, 3 and 4 or 4 and 1. This is because these are the situations which lead to idling will at station 1 under the HIP. When there are no jobs present at station 2, the HIP acts the same as the non-idling policy. Hence the total WIP and therefore the long run average WIP under the HIP are less than or equal to that under the non-idling policy. □

Of course this result is not interesting unless we can show that there exists a case in which the HIP actually causes a reduction in average WIP. This is accomplished by the example in subsection 5.1.1.

Lemma 5.1.1 establishes that the optimal policy with respect to WIP lies in the class of idling policies. Note that the HIP mentioned above may not be the WIP optimal policy in all cases. Under the HIP idling occurs only when the number of jobs removed from the system is guaranteed to be greater than or equal to the number of jobs removed from the system under the non-idling

policy. An interesting question that arises is: Are there any conditions under which idling may be WIP optimal if there are no jobs at station 2? Even under the assumption that all arrivals are processes are Poisson this is not a simple question to answer. In an effort to obtain insight into when idling might be optimal at station 2 we simulated this system under various idling policies using a C program (code listed in Appendix 1). The details of the simulation are outlined below in subsection 5.2.

5.2 Simulation

The system tested is the SVCR under various policies listed in table 5.1. The random number generator used is a combined mixed random number generator written by L'Ecuyer (see Appendix 7B, [1]). The arrival processes at each station are Poisson. The arrival rates at stations 1, 2 and 3 were set at 0.125, 0.1225 and 0.1225 respectively. These arrival rates correspond to 50% utilization at station 1 and 99% at stations 2 and 3 respectively. The vehicle was initially at station 4 and it was assumed that there were no jobs in the system at time 0. The simulation was terminated after four million units of time. This was deemed to be long enough to remove the initialization bias. Thirty replications were performed. The simulation code for the HIP is shown in 1. The policies tested in the SVCR are listed in table 5.1.

The simulation was validated with a simple example via Excel and via the deterministic example mentioned in section 5.1.1. Of all the policies tested under the rates described here, the policy with the lowest overall average WIP

Policy Label	**Policy**
Policy 1	The heuristic idling policy.
Policies 2 - 6	Idle at station 1 whenever there are jobs at station 2. or if the number of jobs at station 1 is below a certain threshold. The threshold at station 1 was varied from 1 to 5.
Policy 7	Non-idling policy
Policy 8	Idle at station 1 whenever there are jobs at station 2 or with probability $1 - e^{-\alpha_{2,1}}$ if there are no jobs at station 2.
Policy 9	Idle at station 1 whenever there are jobs at station 2 or with probability $1 - e^{-\alpha_{2,1}}$ if there are jobs at station 3 but no jobs at station 2.
Policy 10	Idle at station 1 if there are jobs at stations 2 or 3.

Table 5.1: Scheduling policies tested

(mean over thirty replications) was the HIP followed by the non-idling policy and the threshold idling policies (policies 2 - 6). The policies in which idling occurred at station 1 whenever there were no jobs at station 2 (Policies 8 and 9) performed very poorly with respect to WIP. The results on the mean values obtained are summarized in table 5.2.

Looking at just the mean values over thirty replications, the threshold policies simply increase the WIP in the system by the value of the threshold. A possible explanation for this is that the value of holding a number of units of class $(1, 3)$ jobs in the system causes an increase in WIP in the system which outweighs the reduction in WIP obtained by idling. This could be because the probability of arrival in one time unit is low due to the low arrival rates

Scheduling Policy	Average WIP (over 30 Replications)
HIP	94.9244
Policy 2	95.9243
Policy 3	96.9241
Policy 4	97.9241
Policy 5	98.9240
Policy 6	99.9240
Policy 7	95.8672
Policy 8	21178.23
Policy 9	18595.59
Policy 10	101233.83

Table 5.2: Simulation Test Results

chosen in order to maintain stability. Large values of the threshold were also investigated and analogous results were obtained. We have not included the results from those thresholds here. The probabilistic idling policies (Policies 8 and 9) were found to be extremely far from optimal with respect to WIP. They cause the average WIP in the system to increase with time. Another idling policy (Policy 10) was also found to perform very poorly in terms of average WIP levels. Hence policies 8, 9 and 10 were excluded from the group means tests performed.

5.2.1 Group Means Tests

The average WIP from the replications for each of the policies 1-6 was analyzed using JMP at a significance level of 0.1. The tests performed were student's t-distribution between each pair and Tukey-Kramer Honestly Significant Dif-

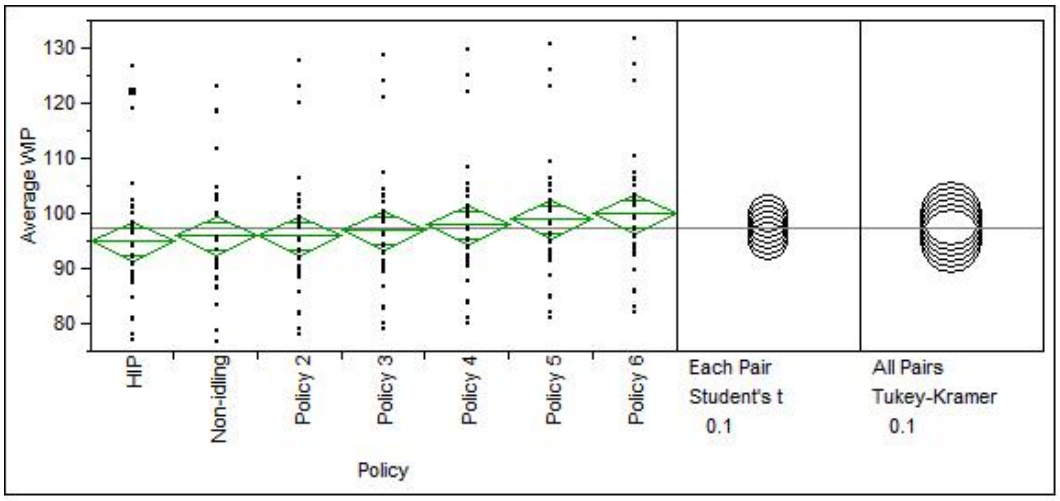

Figure 5.3: Group Means Test

ference (HSD) tests. The result of these tests are shown in Figure 5.3. The Tukey-Kramer HSD test classified all the policies as belonging to the same group. In other words there was no significant difference between the means at a 0.1 alpha level. However under the each pair student's t-test, the difference in means between policies 1 and 6 was significant. The table of t-values obtained is shown in table 5.3. Positive values show pairs of means that are significantly different. Hence the only pair which has a significant difference is the pair (Policy 6, HIP).

Hence under the conditions in which this simulation was conducted we find that the value of idling, even though it led to a decrease in average WIP, is very limited and does not make a significant difference. The complete results from the tests including the confidence intervals are provided in appendix 2. While this simulation study yielded valuable insight into when idling may be

Abs(Dif)-LSD	Policy 6	Policy 5	Policy 4	Policy 3	Policy 2	Non-idling	HIP
Policy 6	-4.9710	-3.9710	-2.9711	-1.9711	-0.9713	-0.9142	0.0286
Policy 5	-3.9710	-4.9710	-3.9710	-2.9710	-1.9712	-1.9142	-0.9713
Policy 4	-2.9711	-3.9710	-4.9710	-3.9710	-2.9712	-2.9141	-1.9713
Policy 3	-1.9711	-2.9710	-3.9710	-4.9710	-3.9711	-3.9141	-2.9712
Policy 2	-0.9713	-1.9712	-2.9712	-3.9711	-4.9710	-4.9139	-3.9710
Non-idling	-0.9142	-1.9142	-2.9141	-3.9141	-4.9139	-4.9710	-4.0281
HIP	0.0286	-0.9713	-1.9713	-2.9712	-3.9710	-4.0281	-4.9710

Table 5.3: Simulation Test Results

optimal, there are still a lot of other policies which can be tested. Furthermore, the system tested was relatively small. This simulation can be extended to systems with any number of vehicles (less than or equal to the number of stations). The performance of more complex systems (described in chapter 6) can also be studied using simulation.

5.3 Idling Example for GFCR

In this section we provide an example of a situation in which an idling policy is better than any non-idling policy with respect to WIP for the GFCR system. Again we limit our consideration to non-ejective policies.

Theorem 5.3.1. *Under the class of non-ejective policies there exists a non-optimal non-idling policy.*

Proof. We prove this theorem by means of a deterministic example. Consider a GFCR with four stations to which arrivals occur and a destination station 5, as shown in figure 5.4. External arrivals occur only to classes $(1,4),(2,1),(3,1),(4,1)$. We assume that class $(1,4)$ has one job in the buffer at time 0. All the other buffers are assumed to be empty initially. Each of these classes receives a deterministic external arrival stream with arrival rates of 0.25 jobs per unit time and arrival times as described below:

- Class $(1,4)$ receives external arrivals at times $4, 8, 12, \ldots$.
- Class $(2,1)$ receives external arrivals at times $1, 5, 9, \ldots$.

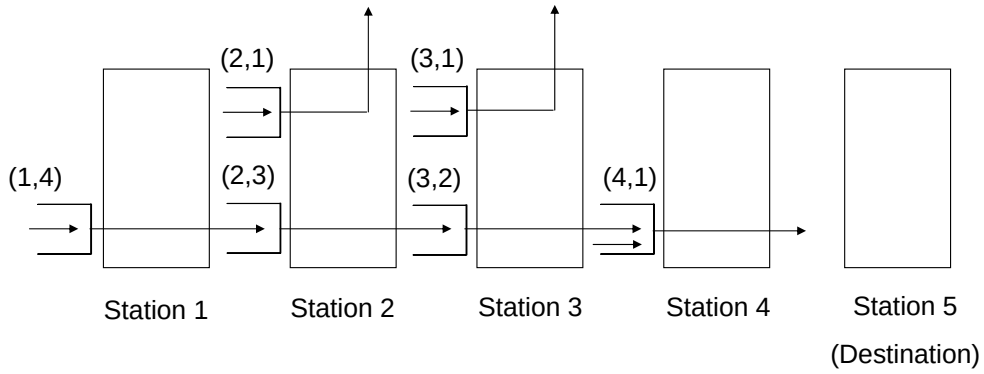

Figure 5.4: Idling Example

- Class $(3, 1)$ receives external arrivals at times $2, 6, 8, \ldots$.
- Class $(4, 1)$ receives external arrivals at times $3, 7, 11, \ldots$.

We assume that all the vehicles in the GFCR are empty at time 0. The only choice between idling and non-idling occurs at station 1. There is nothing to be gained by idling at any of the other stations as they do not "block" any jobs at the downstream stations. At every integer unit of time we know that an empty vehicle arrives at station 1. The only non-idling policy in this case is to load

a job waiting at station 1 into an empty vehicle at times $0, 4, \ldots$. The idling policy that we consider is one in which we idle at time units $0, 4, \ldots$. That is, we let the vehicles that arrive at station 1 at time units $0, 4, \ldots$ travel empty to station 2 even though there are jobs waiting at station 1. We then load a job waiting at station 1 into the vehicle that arrives at time units $1, 5, \ldots$.

Under the non-idling policy, at time 0, there is just one job in the GFCR. This job is at station 1. At time 1, this job is transferred to station 2. But at station 2, there is an arrival at time 1. This arrival cannot be loaded on the vehicle as it is occupied. Hence this arrival must wait until the next time unit to be loaded onto a vehicle. Hence there are two jobs in the GFCR during the interval $[1, 2)$. At time 2, this job is then transported to station 3, where an arrival has just occurred. Hence at time 2, there are three jobs waiting for service (one belonging to class $(2, 1)$, one belonging to class $(3, 1)$ and one belonging to class $(3, 2)$). Service on job belonging to class $(3, 2)$ commences at this time. At time 3, there are three jobs (one belonging to class $(3, 1)$, and two belonging to class $(4, 1)$ as an external arrival occurs to class $(4, 1)$). Note that the job present initially at station 1 belongs to class $(4, 1)$ at time 3. At time 4, the job belonging to class $(3, 1)$ and one job of class $(4, 1)$ exit the system. Hence there are two jobs in the system at time 4. One at class $(4, 1)$ and one at class $(1, 4)$. At time 5, the job of class $(4, 1)$ exits the system and the job of class $(1, 4)$ is at station 2. There are two jobs in the system as an external arrival occurs to class $(2, 1)$ at this time. Also note that this state is identical to the state of the system at time 1. Hence from this point

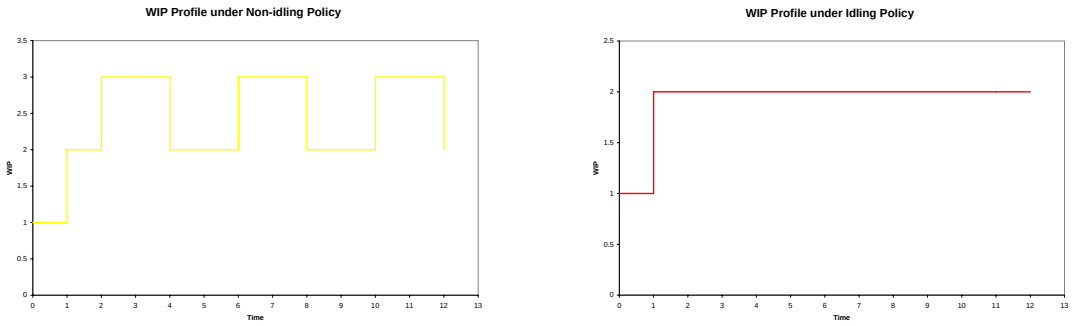

Figure 5.5: WIP Profiles under Idling and Non-Idling policies

of time the WIP oscillates between 2 and 3 units every two units of time and this leads to an average WIP of 2.5 units.

Under the idling policy at time 0, we let the vehicle travel empty to station 2. This enables us to pick up a job at station 2 at time 2. Also at time 2, the job at station 1 is loaded on to an empty vehicle. This idling policy also enables us to pick up a job at station 3 at time 3 and at station 4 at time 4. The WIP in this system is maintained at two units for all times after time 1. The WIP profile for both of these policies is shown in figure 5.5. Thus even in simple versions of a GFCR there are cases in which it might be optimal to idle for a better WIP performance.

□

Chapter 6

Conclusions and Future Research

The flexibility available to model complex systems in applications ranging from manufacturing to communication offers a strong incentive to apply queueing theory to model a particular system. Although the direct analysis of a complex stochastic model is extremely difficult, there exist approximation techniques which provide insight into various aspects of the system such as stability. In this dissertation we have investigated the throughput optimality of a few special cases of the CR. In addition we have focused on optimizing WIP for the fluid model. In this chapter we summarize the work of the previous sections and also outline some interesting questions about this problem that as of now remain unanswered.

The CR was modeled as a unidirectional ring type MQN with an additional operational constraint in this framework. In chapter 3, we described a special case of the CR (the SFCR) and showed that all non-idling policies are throughput optimal in a SFCR. We then proved that for an initial configuration with no jobs at downstream stations all non-idling policies are optimal with respect to WIP minimization in the queueing network. The SFCR system was then represented by the fluid model of the UMQN and SRTT was shown

to be a WIP optimal policy for the fluid network with a general initial configuration. For an initial configuration with no fluid at downstream buffers all policies were optimal which was in accordance with the result for the queueing network.

We then extended this model in chapter 4 to a more general system referred to as the GFCR. Again throughput optimality and WIP optimality were established. For this version of the CR we showed that the SRTT policy was optimal with respect to WIP minimization in the fluid model. However it was shown that a direct application of this policy to the queueing network was not optimal with respect to the WIP minimization. It is well-known that there are many queueing network policies which correspond to a given policy in the fluid network. Maglaras [19] provides an especially illuminating example which demonstrates the difficulty of correctly translating a fluid policy to the queueing network. Thus one avenue to be explored is the mapping of the SRTT policy using the discrete review framework developed by Maglaras to identify optimal policies for the queueing network.

An obvious extension to our work is to determine if all non-idling policies are throughput optimal in the general CR i.e, with the feed-forward restriction removed. That is, the goal would be to determine the stability region for a CR in which any job at any station can request a destination along the ring before its station of arrival. At this point, it is unclear if our proof method can be directly extended, or if a novel argument is required to establish the general case.

We have also shown that in the CR, idling in some situations is optimal with respect to WIP minimization. A CR with a single vehicle and four stations was used as an illustrative example. We showed that for this example a heuristic idling policy always led to a lower long run average WIP than any non-idling policy. Thus we established the result that there exists a non-optimal non-idling non-ejective policy with respect to WIP minimization. We have also presented an example of a GFCR with four vehicles in which idling may be optimal with respect to WIP minimization.

Investigating the effect of vehicle breakdowns is another potential extension of the work of this dissertation. In a way the four station CR with a single vehicle can be viewed as a CR in which three of the vehicles are down permanently. However this is an extreme case. A general case of interest might be one in which vehicle up and down times are sequences of i.i.d. random variables. During the down times the vehicle would travel around the ring but not take on any jobs.

An alternate means of obtaining good scheduling rules could be the use of a diffusion approximation to approximate the queue length process. In this method we consider a sequence of systems with traffic intensity approaching one. Since the traffic intensity at each station increases, the (unscaled) queue length process diverges as the traffic intensity approaches 1. As in the fluid limit process, the queue length process needs to be rescaled. A heavy traffic approximation investigates the limit of the queue length process under the scaling $\frac{Q^r_{i,j}(rt)}{\sqrt{r}}$ as r goes to infinity. Once the diffusion approximation has

been shown to exist for the system and the limiting process determined, one can formulate, and often solve a stochastic control problem. The resulting control policy can then be translated back to the queueing network to obtain a heuristic scheduling policy.

As mentioned in chapter 1, simulation of these systems could be an effective way to develop heuristic policies which are near optimal with respect to WIP minimization or other performance measures. In fact, current work is being done in this direction in Bauer [4], in which he attempts to approximate a "value function" for the associated discrete control problem with a fluid model.

Note that the general CR is only an approximation of an AMHS. More complex models of the AMHS arise by adding "crossovers." In an AMHS with crossovers, vehicles are allowed to take predefined shortcuts through the center of the ring. Synchronizing the vehicles is a difficult task and scheduling problems take on a more interesting dimension. For example if there is a shorter path (than the route along the ring) which connects stations i and $(i+k) \mod N$, then the choice of job which takes this shorter path becomes extremely interesting. For example an empty vehicle can be made to travel on this crossover if a higher priority job is waiting at a station just past the point where the crossover rejoins the ring. Also once a vehicle has been loaded with a job of a particular destination, it can be made to travel on the crossover if it offers a shorter route, thus freeing up the vehicle for another job earlier. Hence even the presence of a single crossover complicates the scheduling problem and offers a much wider variety of interesting scheduling rules. Simulation might

be a good way to obtain insight into systems with crossovers.

Thus even though much effort has been put into analyzing some of the aspects of this problem, many interesting unexplored avenues of research remain.

Appendix

Appendix 1

```
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <stdio.h>
double mrand();

double drng[6] ={ 832983129.0,   865924509.0,  2541312012.0,
1173493945.0,  1683091697.0,  4294843572.0};

/* Main Program */
int main()
{
  double x;
  double arrtime1=0, arrtime2=0, arrtime3=0;
  double rate1, rate2, rate3;
  double * wip=NULL;
  int * q =NULL;
  int i,j,k;
  int n1,n2,n3;
  int p1,p2,p3,r2,s3;
  int rotations,thres;
  char c[1];
  char cand_file1[20];
  char cand_file2[20];
  char cand_file3[20];
  FILE * fp1=NULL;
  FILE * fp2=NULL;
  FILE * fp3=NULL;
  FILE * fp4=NULL;

  wip=(double*)calloc(3,sizeof(double));
  q=(int*)calloc(3,sizeof(int));
```

```
  printf("give the rates\n");
  rate1=.125;rate2=.1225;rate3=.1225;
  thres=0;
  rotations=1000000;

  for(k=0;k<=thres;++k){

  strcpy(cand_file1,"p1_");
  c[0]=(char) (k+48);
  strncat(cand_file1,c,1);
  fp4=fopen(cand_file1,"w");

  drng[0] =832983129.0;   drng[1] =865924509.0;  drng[2]
=2541312012.0;
  drng[3] =1173493945.0;  drng[4] =1683091697.0;  drng[5]
=4294843572.0;
  for(j=0;j<30;++j){
  arrtime1=0; arrtime2=0; arrtime3=0;
  wip[0]=0; wip[1]=0; wip[2]=0;
  q[0]=0; q[1]=0; q[2]=0;

  x=mrand();
  arrtime1+=-log(x)/(rate1);
  x=mrand();
  arrtime2+=-log(x)/(rate2);
  x=mrand();
  arrtime3+=-log(x)/(rate3);


  for(i=0;i<rotations;++i){
    n1=0;n2=0;n3=0;
    p1=q[0];p2=q[1];p3=q[2];r2=q[1];s3=q[2];
    while(arrtime1<=4*(i+1)){
      n1+=1;
      wip[0]+=(4*(i+1)-arrtime1);
      if(arrtime1<=4*i+1) p1+=1;
      x=mrand();
      arrtime1+=-log(x)/(rate1);
    }
```

```
while(arrtime2<=4*(i+1)){
  n2+=1;
  wip[1]+=(4*(i+1)-arrtime2);
  if(arrtime2<=4*i+1) p2+=1;
  if(arrtime2<=4*i+2) r2+=1;
  x=mrand();
  arrtime2+=-log(x)/(rate2);
}
while(arrtime3<=4*(i+1)){
  n3+=1;
  wip[2]+=(4*(i+1)-arrtime3);
  if(arrtime3<=4*i+1) p3+=1;
  if(arrtime3<=4*i+3) s3+=1;
  x=mrand();
  arrtime3+=-log(x)/(rate3);
}

wip[0]+=q[0]*4;
wip[1]+=q[1]*4;
wip[2]+=q[2]*4;
q[0]+=n1;
q[1]+=n2;
q[2]+=n3;

x=mrand();

//This is the heuristic idling policy
if(p2>0 || p1<=k){
  if(r2>0) {wip[1]-=1;q[1]-=1;}
  if(s3>0) q[2]-=1;
}
else if (p1>k)
  q[0]-=1;
/*if (p1>k || p2==0)
  q[0]-=1;
else if(p2>0 ){
  if(r2>0) {wip[1]-=1;q[1]-=1;}
  if(s3>0) q[2]-=1;
}*/
```

```
  }
  printf("%lf,\n",(wip[0]+wip[1]+wip[2])/(rotations*4));
  for(i=0;i<3;++i)
    fprintf(fp4,"%lf,\t",wip[i]/(rotations*4));
  fprintf(fp4,"\n");
  }
  fclose(fp4);
  }
double mrand()
{
  const double m1=4294967087.0, m2=4294944443.0;
  const double norm=1.0/(m1+1);
  int k;
  double p;
  double s10=drng[0], s11=drng[1], s12=drng[2];
  double s20=drng[3], s21=drng[4], s22=drng[5];

  p = 1403580.0 * s11 - 810728.0 * s10;
  k = (int)(p/m1);
  p -= k*m1;
  if (p < 0.0) p += m1;
  s10 = s11;  s11 = s12;  s12 = p;

  p = 527612.0 * s22 - 1370589.0 * s20;
  k = (int)(p/m2);
  p -= k*m2;
  if (p < 0.0) p += m2;
  s20 = s21;  s21 = s22;  s22 = p;


  drng[0] = s10;  drng[1] = s11;  drng[2] = s12;
  drng[3] = s20;  drng[4] = s21;  drng[5] = s22;

  if (s12 <= s22) return ((s12 - s22 + m1) * norm);
  else return ((s12 - s22) * norm);
}
```

Appendix 2

Confidence Intervals

Confidence intervals from one-way ANOVA

Policy	Number	Mean	Std Error	Lower 90%	Upper 90%
HIP	30	94.9244	2.1272	91.409	98.44
Non-idling	30	95.8673	2.1272	92.352	99.38
Policy 2	30	95.9244	2.1272	92.409	99.44
Policy 3	30	96.9242	2.1272	93.409	100.44
Policy 4	30	97.9241	2.1272	94.409	101.44
Policy 5	30	98.9241	2.1272	95.409	102.44
Policy 6	30	99.9240	2.1272	96.409	103.44

Table 2.1: Results from One Way ANOVA

Confidence intervals from each pair Student's t-test

Level	- Level	Difference	Lower CL	Upper CL	p-Value
Policy 6	HIP	4.999590	0.02864	9.970540	0.0981
Policy 6	Non-idling	4.056720	-0.91423	9.027671	0.1790
Policy 6	Policy 2	3.999655	-0.97130	8.970606	0.1852
Policy 5	HIP	3.999644	-0.97131	8.970595	0.1852
Policy 5	Non-idling	3.056775	-1.91418	8.027726	0.3108
Policy 6	Policy 3	2.999852	-1.97110	7.970802	0.3199
Policy 5	Policy 2	2.999710	-1.97124	7.970660	0.3199
Policy 4	HIP	2.999699	-1.97125	7.970649	0.3199
Policy 4	Non-idling	2.056829	-2.91412	7.027780	0.4949
Policy 5	Policy 3	1.999907	-2.97104	6.970857	0.5069
Policy 6	Policy 4	1.999891	-2.97106	6.970841	0.5069
Policy 4	Policy 2	1.999764	-2.97119	6.970715	0.5070
Policy 3	HIP	1.999738	-2.97121	6.970688	0.5070
Policy 3	Non-idling	1.056869	-3.91408	6.027819	0.7257
Policy 4	Policy 3	0.999961	-3.97099	5.970911	0.7399
Policy 5	Policy 4	0.999946	-3.97100	5.970896	0.7399
Policy 6	Policy 5	0.999945	-3.97101	5.970896	0.7399
Policy 2	HIP	0.999935	-3.97102	5.970885	0.7399
Policy 3	Policy 2	0.999803	-3.97115	5.970754	0.7400
Non-idling	HIP	0.942869	-4.02808	5.913820	0.7543
Policy 2	Non-idling	0.057065	-4.91389	5.028016	0.9849

Table 2.2: Results from each pair student's t-test

Confidence intervals from Tukey-Kramer HSD test

Level	- Level	Difference	Lower CL	Upper CL
Policy 6	HIP	4.999590	-3.16521	13.16439
Policy 6	Non-idling	4.056720	-4.10808	12.22152
Policy 6	Policy 2	3.999655	-4.16514	12.16445
Policy 5	HIP	3.999644	-4.16515	12.16444
Policy 5	Non-idling	3.056775	-5.10802	11.22157
Policy 6	Policy 3	2.999852	-5.16494	11.16465
Policy 5	Policy 2	2.999710	-5.16509	11.16451
Policy 4	HIP	2.999699	-5.16510	11.16449

Table 2.3: Results from Tukey-Kramer HSD test

Bibliography

[1] A.M.Law and W.D.Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 2000.

[2] B. Avi-Itzhak. Heavy traffic characterisitics of a circular data network. *Bell Syst. Tech. J.*, 50:2521–2549, 1971.

[3] F. Avram, D. Bertsimas, and M. Ricard. Fluid models of sequencing problems in open queueing networks; an optimal control approach. In F. P. Kelly and R. J. Williams, editors, *Stochastic Networks*, volume 71 of *The IMA volumes in mathematics and its applications*, pages 199–237, New York, 1995. Springer-Verlag.

[4] D. Bauer. *Scheduling the Cambridge Ring*. PhD thesis, The University of Texas at Austin, In Progress.

[5] T. Bonald and D. Down. Stability of mixed generalized Jackson networks. *Operations Research Letters*, 25:131–136, 1999.

[6] H. Chen. Fluid approximations and stability of multiclass queueing networks I: Work-conserving disciplines. *Annals of Applied Probability*, 5:637–665, 1995.

[7] H. Chen and D. Yao. Dynamic scheduling of a multiclass fluid network. *Operations Research*, 41:1104–1115, 1993.

[8] J. G. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *Annals of Applied Probability*, 5:49–77, 1995.

[9] J. G. Dai. Stability of fluid and stochastic processing networks. In *MaPhySto Miscellanea Publication, No. 9*. Centre for Mathematical Physics and Stochastics, 1999.

[10] J. G. Dai and G. Weiss. Stability and instability of fluid models for reentrant lines. *Mathematics of Operations Research*, 21:115–134, 1996.

[11] J.-F. Dantzer and V. Dumas. Stability analysis of the Cambridge Ring. *Queueing Systems: Theory and Applications*, 40:125–142, 2002.

[12] E. G. Coffman Jr., L. Flatto, E. N. Gilbert, and A. G. Greenberg. An approximate model of processor communication rings under heavy load. *Information Processing Letters*, 64:61–67, 1997.

[13] E. G. Coffman Jr., E. N. Gilbert, A. G. Greenberg, F. T. Leighton, P. Robert, and A. L. Stolyar. Queues served by a rotating ring. *Communications in Statistics – Stochastic Models*, 11(3):371–394, 1995.

[14] D. Gamarnik and J. J. Hasenbein. Instability in stochastic and fluid queueing networks. *Annals of Applied Probability*, 15(3):1652–1690, 2005.

[15] J. J. Hasenbein, S. Sigireddy, and R. Wright. Taking a queue from simulation. *Industrial Engineer*, 36(8):39–43, 2004.

[16] A. Hopper and R. M. Needham. The Cambridge Fast Ring networking system. *IEEE Transactions on Computers*, 37(10):1214–1223, 1988.

[17] P. J. B. King and I. Mitrani. Modelling The Cambridge Ring. *Proceedings of the 1982 ACM SIGMETRICS conference on Measurement and modeling of computer systems*, pages 250–258, 1982.

[18] S. H. Lu and P. R. Kumar. Distributed scheduling based on due dates and buffer priorities. *IEEE Transactions on Automatic Control*, 36:1406–1416, 1991.

[19] C. Maglaras. Discrete-review policies for scheduling stochastic networks: trajectory tracking and fluid-scale asymptotic optimality. *The Annals of Applied Probability*, 10(3):897–929, 2000.

[20] R. M. Needham. System aspects of The Cambridge Ring. *Proceedings of the seventh ACM symposium on Operating systems principles*, pages 82–85, 1979.

[21] A. N. Rybko and A. L. Stolyar. Ergodicity of stochastic processes describing the operation of open queueing networks. *Problems of Information Transmission*, 28:199–220, 1992.

[22] G. Weiss. On optimal draining of fluid reentrant lines. In F. P. Kelly and R. J. Williams, editors, *Stochastic Networks*, volume 71 of *The IMA volumes in mathematics and its applications*, pages 93–105, New York, 1995. Springer-Verlag.

[23] G. Weiss. Optimal draining of fluid reentrant lines: some solved examples. In F. P. Kelly, S. Zachary, and I. Zeidins, editors, *Stochastic Networks: Theory and Applications*, volume 4 of *Royal Statistical Society Lecture Note Series*, pages 19–34, Oxford, England, 1996. Oxford University Press.

[24] G. Weiss. Scheduling and control of manufacturing systems – a fluid approach. In *Proceedings of 37th Allerton Conference, Monticello, Illinois*, pages 557–586, Sept 1999.

[25] G. Weiss. A simplex based algorithm to solve separated continuous linear programs. *Mathematical Programming A.*, 2005. Submitted.

Vita

Balaji Sampath was born on the 21^{st} of November, 1977 as the eldest son of Mrs. Sugantha Sampath and Dr. S Sampath. He attended and graduated from the University of Madras with a baccalaureate degree in Mechanical Engineering. He joined graduate school at the University of Texas at Austin in Fall 1999. After completing his Master of Science in Manufacturing Systems Engineering in August 2001, he continued his graduate studies in Operations Research with a view to obtain his Ph.D.

Permanent address: 3501 Speedway, apt.103
Austin, Texas 78705

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.